IMAGE SEGMENTATION WITH HIERARCHICAL MODELS

by

Ting Liu

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computing

School of Computing

The University of Utah

August 2016

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Ting Liu**

has been approved by the following supervisory committee members:

Tolga Tasdizen	, Chair	**4/29/2016** Date Approved
P. Thomas Fletcher	, Member	**4/29/2016** Date Approved
Guido Gerig	, Member	**5/3/2016** Date Approved
Bryan W. Jones	, Member	**5/9/2016** Date Approved
Jeffrey M. Phillips	, Member	**4/29/2016** Date Approved

and by **Ross T. Whitaker**, Chair/Dean of

the Department/College/School of **Computing**

and by David B. Kieda, Dean of The Graduate School.

ABSTRACT

Image segmentation is the problem of partitioning an image into disjoint segments that are perceptually or semantically homogeneous. As one of the most fundamental computer vision problems, image segmentation is used as a primary step for high-level vision tasks, such as object recognition and image understanding, and has even wider applications in interdisciplinary areas, such as longitudinal brain image analysis. Hierarchical models have gained popularity as a key component in image segmentation frameworks. By imposing structures, a hierarchical model can efficiently utilize features from larger image regions and make optimal inference for final segmentation feasible.

We develop a hierarchical merge tree (HMT) model for image segmentation. Motivated by the application in large-scale segmentation of neuronal structures in electron microscopy (EM) images, our model provides a compact representation of region merging hypotheses and utilizes higher order information for efficient segmentation inference. Taking advantage of supervised learning, our model is free from parameter tuning and outperforms previous state-of-the-art methods on both two-dimensional (2D) and three-dimensional EM image data sets without any change. We also extend HMT to the hierarchical merge forest (HMF) model. By identifying region correspondences, HMF utilizes inter-section information to correct intra-section errors and improves 2D EM segmentation accuracy.

HMT is a generic segmentation model. We demonstrate this by applying it to natural image segmentation problems. We propose a constrained conditional model formulation with a globally optimal inference algorithm for HMT and an iterative merge tree sampling algorithm that significantly improves its performance. Experimental results show our approach achieves state-of-the-art accuracy for object-independent image segmentation.

Finally, we propose a semi-supervised HMT (SSHMT) model to reduce the high demand for labeled data by supervised learning. We introduce a differentiable unsupervised loss term that enforces consistent boundary predictions and develop a Bayesian learning model that combines supervised and unsupervised information. We show that with a very small amount of labeled data, SSHMT consistently performs close to the supervised HMT with full labeled data sets and significantly outperforms HMT trained with the same labeled subsets.

To my parents.

CONTENTS

LIST OF FIGURES

Figures

LIST OF TABLES

Tables

ACKNOWLEDGMENTS

First, I would like to thank my advisor Dr. Tolga Tasdizen for his invaluable guidance and support. His knowledge and inspiration set me off on this quest and guided me through. He showed me how to be an independent researcher and think outside the box. I could not be more grateful for having his advice and being his student. I also want to thank my committee members, my mentors, Dr. P. Thomas Fletcher, Dr. Guido Gerig, Dr. Bryan W. Jones, and Dr. Jeff M. Phillips, who have been of tremendous help with their insight and expertise from different areas.

This research was made possible with help from many people at the University of Utah. I would like to thank my fellow students and researchers, including Dr. Elizabeth Jurrus, Dr. Mojtaba Seyedhosseini, Dr. Cory Jones, Mehdi Sajjadi, Mehran Javanmardi, Nisha Ramesh, Dr. Paul Rosen, Yang Gao, Dr. Bo Wang, Dr. Wei Liu, Dr. Xiang Hao, Dr. Zhisong Fu, Dr. Qingyu Meng, Dr. Aaron Knoll, Dr. Xueyu Zhu, Dr. Yanyan He, Hang Shao, Dr. Miaomiao Zhang, and many others, who helped me at different points in my Ph.D. career with enthusiastic discussions and constructive comments. Much appreciation goes to the faculty and staff at the School of Computing. I must thank every professor who imparted their knowledge and helped pave my way for being a researcher. A big thank you to the leadership and support teams at the Scientific Computing and Imaging Institute for providing us with great resources and a home-like environment.

I would like to acknowledge the National Institute of Health and the National Science Foundation for the funding support, NIH 1R01NS075314-01 and NSF IIS-1149299, to the work in this dissertation. I also thank our collaborators in Dr. Mark Ellisman's group at the National Center for Microscopy and Imaging Research, University of California, San Diego, for their discussions and feedback, which helped improve our work.

To all my friends, thank you for the wonderful moments we shared together, which I will always remember. Portia, thank you for being there for me with all the understanding, encouragement, and support when I needed them most. Finally, I owe my deepest gratitude to my parents. It is your love that made this all worthwhile. It is to you that I dedicate this work.

CHAPTER 1

INTRODUCTION

This dissertation focuses on developing learning-based frameworks for hierarchical image segmentation. Image segmentation considers an image as a set of pixels and seeks to find a partition of this set into multiple disjoint subsets, such that pixels in each subset share certain visual or semantic characteristics. General image segmentation is widely used as a preprocessing step for solving high-level vision problems, such as object recognition and image understanding [1, 2]. In many interdisciplinary areas, such as biological and medical image analysis, image segmentation also plays an important role in helping scientists quantify and analyze image data [3, 4].

There are two perspectives of image segmentation [5]: edge detection and region segmentation. Edge detection aims at finding edges between different perceptual pixel groups. Region segmentation partitions an image into disjoint regions. Usually, edge detection focuses on assigning a binary label to each pixel with certain confidence, indicating if it belongs to an edge or not, and does not guarantee closed object contours. Though closed contours and the regions they encircle can be recovered from edges, such transformation with high accuracy is usually nontrivial. On the other hand, region segmentation seeks to find the cluster membership of each pixel, and closed contours of an object can be trivially generated as the outmost points of a region. Many region segmentation methods take advantage of the edge detection outputs as boundary cues to help with the search for correct partitioning. The works in this dissertation belong to the region segmentation category.

Region segmentation is a pixel clustering problem subject to the connectivity constraint that there must exist a connected path between any pair of pixels in the same cluster. The first question is at what granularity we do the clustering. Pixel-based methods have been proposed in early works [6, 7, 8] and suffer from two major problems. First, the computational complexity of such methods is usually very high as in the order of the number of all pixels. Second, individual pixel intensities are usually unable to describe image context.

For example, it is not possible to segment neuron cells in the electron microscopy image shown in Figure 1.1a by looking only at each pixel, because the intracellular structures, such as mitochondria and vesicles, have intensities very similar to those of the membranes. The use of superpixels offers a solution. Superpixels, with an example shown in Figure 1.1c, are (small) perceptually or semantically homogeneous image regions that are near-complete: they presumably always preserve image structures and do not undersegment desirable image objects [9]. Therefore, segmenting image objects becomes correctly identifying and clustering their superpixels. Using superpixels as segmentation primitives largely reduces the computational complexity to the order of the number of superpixels. Moreover, informative nonlocal features can be extracted to guide the segmentation process.

Another important question is how to do the clustering based on image context. There is a trade-off between the utilization of higher order image information and tractability. A

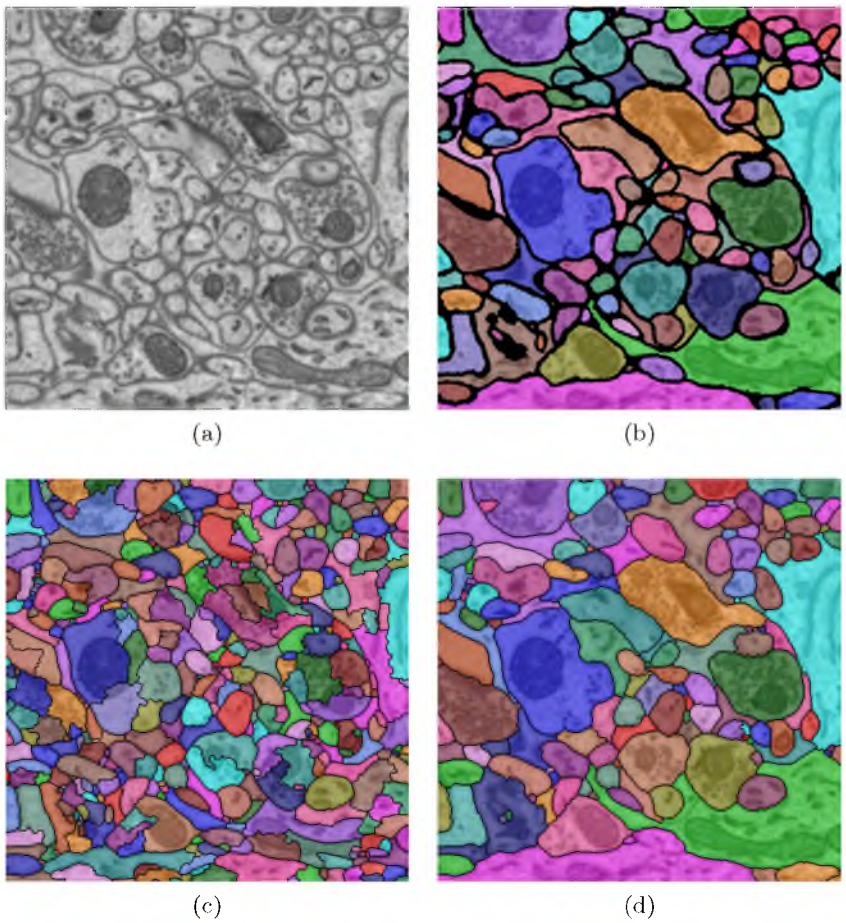


Figure 1.1: Example of (a) an original EM image, (b) a ground truth segmentation, (c) an initial superpixel segmentation, and (d) a proposed final segmentation as a combination of initial superpixels. Different colors indicate individual connected components.

common approach is to consider segmentation as a planar graph labeling problem, which can be solved efficiently under the Markov assumption [10, 11, 12]. Such approaches, however, can use features only from initial superpixels, which may not be meaningful given their small sizes. On the other hand, an exhaustive search over all possible superpixel combinations is intractable. As a trade-off solution, greedy region agglomeration [13] can use features from larger regions by updating them after each merge, but the merging termination is based on local criteria and may not be accurate. Hierarchical segmentation, inspired by hierarchical clustering [14], provides a way to incorporate higher order information and infer final segmentation in a globally optimal manner by imposing structures on the region merging process. In this dissertation, we introduce a tree-like hierarchical image segmentation model and its variants that take advantage of supervised/semi-supervised learning techniques. We apply them to two-dimensional (2D) and three-dimensional (3D) electron microscopy image segmentation for neural circuit reconstruction and 2D natural image segmentation problems and demonstrate state-of-the-art results. Examples of results are shown in Figure 1.1 and 1.2.

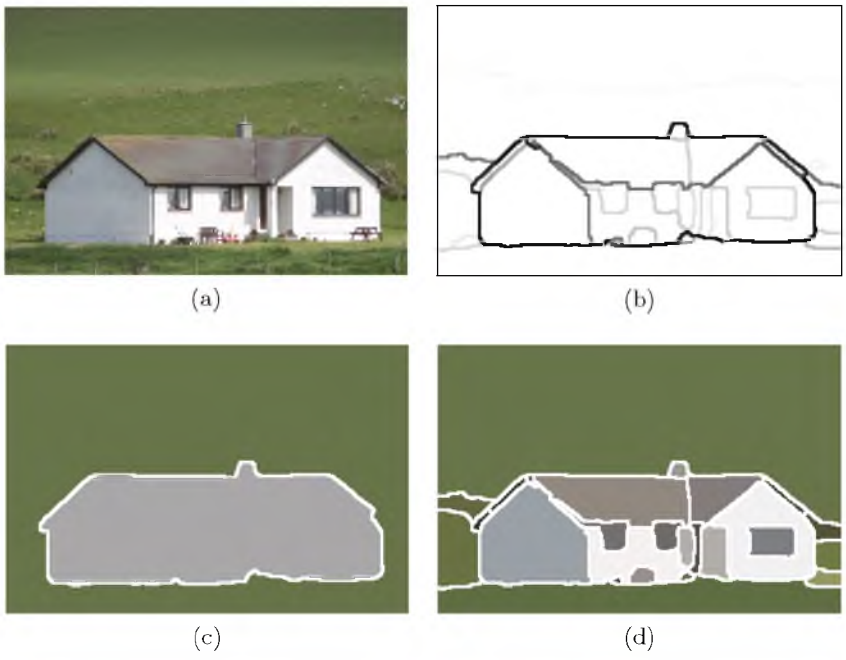


Figure 1.2: Example of (a) an original natural image, (b) a contour hierarchy image, (c) a segmentation at threshold 0.6, and (d) a segmentation at threshold 0.1. Region boundaries are thickened for visualization purposes.

1.1 Neural Circuit Reconstruction

Our works are motivated by the applications of neural circuit reconstruction in connectomics research. Connectomics [15] is drawing attention in neuroscience as an important method for studying neural circuit connectivity and the implied behaviors of nervous systems [16, 17, 18]. It has also been shown that many diseases are highly related to abnormalities in neural circuitry. For instance, changes in the neural circuitry of the retina can lead to corruption of retinal cell class circuitry, and therefore retinal degenerative diseases can be found by ultrastructural cell identity and circuitry examination, which at the same time implies certain strategies for vision rescue [19, 20, 21, 22].

To study the connectivity of a nervous system, image analysis techniques are widely adopted as an important approach. For image acquisition, electron microscopy (EM) provides sufficiently high resolution on nanoscale to image not only intracellular structures but also synapses and gap junctions that are required for neural circuit reconstruction. The EM image data sets we use for neuron segmentation are acquired, respectively, using serial section transmission electron microscopy (SSTEM) [23, 24], serial block-face scanning electron microscopy (SBFSEM) [25], serial section scanning electron microscopy (SSSEM) [26], and focused ion beam scanning electron microscopy (FIBSEM) [27]. Depending on the type of technique, the imaging resolutions range from 2 to 50 nm/pixel, and the EM image data sets for even very small tissue specimens are on terabyte scale [28]. Dense manual analysis is thus extremely laborious and can take decades to complete [29]. Therefore, automatic image-based connectome reconstruction techniques that can extensively reduce human workloads are required. Currently, fully automatic EM image segmentation and connectome reconstruction remain a challenging problem because of the complex ultrastructural cellular textures and the considerable variations in shape and physical topologies within and across image sections [30, 31]. Our work focuses on neuron segmentation as a first step for automating connectomics and can be extended to segmentation of intracellular structures, such as mitochondria.

Our goal is to transform an EM image section/volume into a 2D/3D label map, in which pixels belonging to the same neuron cell have identical labels. Our proposed methods take advantage of ground truth data labeled by human experts and build tree-like structures for supervised hierarchical segmentation of unlabeled image data. We also develop a semi-supervised model that can be accurately trained with very little labeled data. Our supervised methods set new marks on the state-of-the-art performance and achieve even close-to-human segmentation accuracy on certain data sets. Our semi-supervised method

enables accurate segmentation of neuronal structures with very sparse human labeling. Our methods promise a great potential to facilitate the process of reconstructing neuronal structures from EM images and help neuroscientists understand structures and functions of nervous systems.

1.2 Natural Image Segmentation

Different from image segmentation problems for specific tasks that always have one gold standard answer, such as neuron segmentation in EM images, segmentation of a natural image can have multiple "true" answers, each of which describes the image content at a different level of perceptual granularity or semantic organization. For example, when segmenting an image of a keyboard, one segmentation considers the entire keyboard as a single segment, whereas another segmentation labels every key as an individual segment. Both segmentations are "correct." Then, the question becomes how to measure the "correctness" of the segmentations of a natural image. Although opinion exists that segmentation quality can be evaluated only in the context of a specific task [32], the argument by Martin et al. is widely accepted that a segmentation per se can be evaluated via comparisons with multiple ground truth answers generated by human observers [33]. Instead of producing one single label map, a hierarchy of closed contours with different magnitudes should be proposed for an image, which can be trivially transformed to label segmentations at different granularity levels via thresholding.

The content of a natural image can be arbitrary. It ranges from a still animal to a moving car, or from portraits of a single human to views of a crowded street, etc. Moreover, photographs from cameras, a major modality for imaging natural scenes, are subject to inconsistent qualities and various issues, such as noises, chromatic aberrations, image distortions, unfocused exposures, etc. Therefore, natural image segmentation is in general a much more difficult problem than segmentation of specific types of images, and the performance of state-of-the-art methods is still less than satisfactory after years of active research.

We reformulate and extend our hierarchical model for EM segmentation to cope with the issues imposed by the difficult nature of natural image segmentation and to generate the contour hierarchies suggested by [33]. Our model requires no semantic information about an image. We show on a variety of publicly available natural image data sets that our method achieves state-of-the-art performance and is very competitive in general object-independent image segmentation. An example of the resulting contour hierarchy and segmentations at

two threshold levels to capture different details is shown in Figure 1.2.

1.3 Contributions

The contributions of this dissertation include:

1. Development of a hierarchical merge tree (HMT) model for EM image segmentation. The main advantage of this model is that the merge tree structure provides a most efficient representation of potential merging of initial superpixels and is capable of incorporating higher order information to infer final segmentations. Moreover, the proposed model uses supervised learning to take advantage of ground truth data and is almost parameter-free given the initial superpixels. The model is independent of image dimensionality and can thus be applied to both 2D and 3D segmentation without any change. This work is discussed in detail in Chapter 2.

2. Development of a hierarchical merge forest (HMF) model for 2D EM image segmentation. The HMF model improves the performance of the HMT model by utilizing inter-section information. By identifying correspondences between pairs of 2D regions that belong to the same 3D neuron cell, the HMF model couples individual HMT models for each 2D section into a joint model and globalizes the inference of final segmentations of each 2D section in a 3D image volume. This work is discussed in detail in Chapter 3.

3. Development of a constrained conditional model formulation for the HMT model (CCHMT) and an iterative merge tree sampling and segmentation accumulation algorithm for natural image segmentation. The constrained conditional model formulation of HMT enables fast computation of a globally optimal solution. The iterative algorithm efficiently explores the merge tree space and diversifies the merge tree generation. The segmentation accumulation procedure emphasizes accurate boundaries and phases out nonsystematic errors. This work is discussed in detail in Chapter 4.

4. Development of a semi-supervised hierarchical merge tree (SSHMT) model. Learning the boundary classification function is essential to the performance of the HMT model. We propose a differentiable unsupervised loss term to exploit structural information and limit the search space for such functions by expressing the consistency constraint imposed by the merge tree structure in a relaxed disjunctive normal form. We then propose a Bayesian framework that can incorporate unsupervised information and a very small amount of supervised samples to accurately learn the boundary

classification function. We demonstrate that SSHMT can be trained with significantly less labeled data to segment EM images, for which the ground truth generation requires extensive human effort and expertise. This work is discussed in detail in Chapter 5.

1.4 Software

We have released our code to help other researchers understand and build upon our works. All of our code is written in C++ with dependencies on the Insight Toolkit [34], Eigen [35], and the Boost Libraries [36]. Most of our implementations are highly efficient. For example, it takes less than 4 minutes to run a holistic experiment (including data generation, training, and testing) with the HMT model (Section 2.2) over the entire *Drosophila* VNC data set [37, 38] for the ISBI 2012 EM Segmentation Challenge [39] (Section 2.3.1) on an Apple MacBook Pro laptop computer with 2 GHz Intel Core i7 CPU using one core with single thread and limited memory. The code repository is available at `https://github.com/tingliu/glia`.

1.5 Overview

Chapter 2 gives a mathematical overview of the image region segmentation problem and explains the motivation for using superpixels. We then summarize the related works on EM image segmentation and introduce the HMT model in detail. The superior performance of the HMT model is demonstrated on three EM image data sets.

Chapter 3 introduces the HMF model. We first review the related segmentation methods that use inter-section information for EM image segmentation. Then, we describe the HMF model in detail and show its improvement compared to the HMT model on two EM image data sets.

Chapter 4 first reviews the existing methods for general image segmentation. We then introduce the CCHMT and the optimal inference algorithm. Next, we propose the iterative training and testing algorithm that can be used to diversify merge tree generation and improve the overall results. We show the state-of-the-art performance of our approach on six publicly available natural image segmentation data sets.

Chapter 5 discusses the SSHMT model. We first review learning-based methods for EM image segmentation and discuss the relation between our method and another very recent active learning framework for the same task. We then introduce the SSHMT learning model with the novel unsupervised loss term. On three EM image data sets, we demonstrate the effectiveness of our approach with a very limited amount of supervised data.

CHAPTER 2

HIERARCHICAL MERGE TREE MODEL

Hierarchical image segmentation methods impose structures to incorporate higher order information about shapes and image appearance. Meanwhile, supervised learning-based methods utilize ground truth data and learn to make decisions for complex situations. We combine the merit of hierarchical segmentation and supervised learning and propose the hierarchical merge tree (HMT) model that uses tree structures and boundary classification for region-based image segmentation, which is also presented in [40, 41]. We focus on applying the HMT model to neuron segmentation in EM images in this chapter and will extend it to solve general image segmentation problems later. Starting with a cell boundary detection confidence map, we generate initial superpixel segmentation of the image and build a merge tree structure to represent the region merging hierarchy. A boundary classifier is learned to predict likelihood scores for potential merges in the tree based on nonlocal image features. We then define a potential score for each tree node and infer the final segmentation from the tree merge under a consistency constraint. We validate our approach with three EM image data sets and demonstrate that our approach can achieve close-to-human 2D segmentation accuracy and is highly competitive among various other state-of-the-art methods for both 2D and 3D segmentation. Independent of image dimensionalities and classification models, our approach proposes a general framework for efficient hierarchical image segmentation.

2.1 Introduction

There are two general approaches to fully automatic 3D neuron segmentation and reconstruction using EM images. One approach focuses on segmenting neurons in 2D images and making inter-section linkings for 3D reconstruction. As for 2D neuron segmentation, several unsupervised attempts have been made. Anisotropic directional filtering is applied to enhance membrane continuity [42, 43], but fails to detect membranes with sufficient accuracy, and it cannot remove intracellular structures. Kumar et al. [44] introduced radon-like features to suppress undesirable intracellular structures, but this approach can

achieve only moderate accuracy performance. On the other hand, supervised learning methods have proven successful in detecting membranes and segmenting neurons in 2D. Mishchenko [45] uses neural networks with one hidden layer and Hessian eigenspace features to classify pixels as membrane or nonmembrane. Deep neural networks [30, 46, 47, 48, 49] have been widely used for membrane detection and achieved remarkable results. Laptev et al. [50] use SIFT flow to align adjacent image sections and incorporate both intra- and inter-section pixel information for membrane detection. Seyedhosseini et al. [51] propose a multiclass multiscale series contextual model that utilizes both intra- and interobject information within a serial classifier framework for the detection of membranes and other cellular structures simultaneously. Membrane detection results in cell boundary confidence probability maps, which can be simply thresholded [39] to acquire region segmentation. Other more sophisticated methods are proposed to improve the region accuracy. Kaynig et al. [52] propose a graph-cut framework with perceptual grouping constraints to enhance the closing of membrane gaps. For 3D linking given 2D segmentation, Yang and Choe [53] propose a graph-cut framework to trace 2D contours in 3D. Kaynig et al. [54] exploit geometrical consistency constraints and use the expectation maximization algorithm to optimize the 3D affinity matrix. Vitaladevuni and Basri [55] consider the 3D linking as coclustering each pair of adjacent sections and formulate it as a quadratic optimization problem.

The other group of methods seeks to achieve 2D segmentation and 3D reconstruction at the same time. Andres et al. [56] propose a graphical model framework to incorporate both supervoxel face and boundary curve information for 3D supervoxel merging. Vazquez-Reina et al. [57] generate multiple 2D segmentation hypotheses and formulate the 3D segmentation fusion into a Markov random field framework. Similarly, Funke et al. [58] use tree structures to represent 2D segmentation hypotheses and achieve 2D segmentation and 3D reconstruction simultaneously by solving a constrained integer linear programming problem. Jain et al. [59] and Nunez-Iglesias et al. [60] use reinforcement learning frameworks to learn merging policies for superpixel agglomeration.

We develop a hierarchical segmentation approach that is independent of image dimensionalities, which is suitable for segmenting both 2D and 3D images. Independent of a specific membrane detection algorithm, our method takes membrane probability maps as input for superpixel generation and uses a hierarchical merge tree structure to represent the merging of multiple region hypotheses. We use supervised classification techniques to quantify the likelihoods of the hypotheses, based on which we acquire region segmentation

via constrained optimization. We show that our method achieves results highly competitive with other state-of-the-art methods. Our 2D segmentation performance on the public EM 2D segmentation challenge data set is even on the human level. Compared with the other region merging method by Nunez-Iglesias et al. [60], the use of node potential in our method (see Section 2.2.4) utilizes higher order information to make merging decisions rather than only thresholding boundary classifier output. In addition, our merge tree framework makes it convenient to incorporate prior knowledge about segmentation, which may further improve the performance.

2.2 Methodology

Given an image I consisting of pixels $\mathcal{P}$, a segmentation is a partition of $\mathcal{P}$, denoted as $S = \{s_i \in 2^{\mathcal{P}} \,|\, \cup_i s_i = \mathcal{P}; \forall i \neq j, s_i \cap s_j = \varnothing\}$, where $2^{\mathcal{P}}$ is the power set of $\mathcal{P}$. A segmentation assigns every pixel an integer label that is unique for each image object. Each s_i, which is a connected subset of pixels in $\mathcal{P}$, is called a segment or region. All possible partitions form a segmentation space $\mathcal{S}_{\mathcal{P}}$. A ground truth segmentation $S_g \in \mathcal{S}_{\mathcal{P}}$ is usually generated by humans and considered as the gold standard. The accuracy of a segmentation S is measured based on its agreement with S_g. In a probabilistic setting, solving a segmentation problem is formulated as finding a segmentation that maximizes its posterior probability given the image as

$$S^* = \arg\max_{S \in \mathcal{S}_{\mathcal{P}}} P(S \,|\, I). \tag{2.1}$$

2.2.1 Initial segmentation

The current trend to alleviate the difficulty in the pixelwise search for S^* is to start with a set of oversegmenting superpixels. A superpixel is an image segment consisting of pixels that have similar visual characteristics. A number of algorithms [7, 13, 61, 62, 63] can be used to generate superpixels. Our methods use, but are not limited to, the watershed transform [64, 65] over some boundary confidence maps generated by membrane detection algorithms [46, 66]. The basic idea of the watershed transform is to consider an image as a terrain map with pixel intensities representing heights. As the rain falls onto the terrain, water flows down along the steepest path and forms lakes in the basins, which correspond to regions or segments of the image, and the borders between the lakes are called watershed lines. In terms of implementation, local minima of the image are used as seeds, from which regions are grown based on intensity gradients until the region boundaries touch. Figure 1.1c shows an example of an initial superpixel segmentation. Our goal is to combine

those superpixels into a final segmentation, shown in Figure 1.1d. In practice, we blur the membrane detection probabilities with a Gaussian filter and ignore the local minima with dynamics below threshold θ_w to avoid too many initial regions. Meanwhile, by using a small value of θ_w, we can ensure oversegmentation.

Let S_o be the initial oversegmentation given by the superpixels. The final segmentation consisting only of merged superpixels in S_o can be represented as $S = \{s_i \in 2^{\mathcal{P}} \mid \cup_i s_i = \mathcal{P}; \forall i \neq j, s_i \cap s_j = \varnothing; \forall i, \exists S' \in 2^{S_o}, \text{s.t. } s_i = \cup_{s'_j \in S'} s'_j\}$. Therefore, the search space for S is largely reduced to $\mathcal{S} \subseteq \mathcal{S}_{\mathcal{P}}$. Even so, however, an exhaustive search is still intractable, and some kind of heuristic has to be injected. We propose to further limit $\mathcal{S}$ to a set of segmentations induced by tree structures and make the optimum search feasible.

2.2.2 Merge tree

First, we define the boundary between two regions s_i and s_j as the set of pixels that are in connected neighborhoods of pixels from both regions

$$\mathcal{B}(s_i, s_j) = N_{n_c}(s_i) \cap N_{n_c}(s_j), \tag{2.2}$$

where $N_{n_c}(s.)$ represents the set of pixels that are n_c-connected to any pixel in region $s.$. The choice of n_c may differ by implementations. In our work, we use $n_c = 4$ for 2D pixel connectivity and $n_c = 6$ for 3D pixel connectivity. If $\mathcal{B}(s_i, s_j)$ is not the empty set, we say that regions s_i and s_j are neighbors. We then define the merge of N_m disjoint regions $\{s_i\}_{i=1}^{N_m}$ as the union of their pixels, which results in a region $s = \cup_{i=1}^{N_m} s_i$. As an example shown in Figure 2.1, the merge of region s_1 and s_2 forms region s_3. Clearly, merging regions eliminates the boundaries between them.

We define a merging saliency function $f_s : S^{N_m} \to \mathbb{R}$ that takes a set of m regions and uses pixel information from the original image and/or boundary confidence maps to determine the merging saliency of the regions as a real scalar. Higher saliency indicates the regions are more likely to merge. In practice, we consider merging only two regions ($N_m = 2$) each time. To determine the merging saliency, we find that the boundary confidence performs more accurately and consistently than the original image intensity. As an example shown in Figure 2.1a, some intracellular structures, e.g., mitochondria, look even darker than the membranes in the original image, and thus using these original intensities could give false indications about region merging saliency. In the membrane detection probability map (Figure 2.1b), however, the strengths of such pixels are suppressed and the membrane pixels are distinguished. Also, we find that the median of the membrane probabilities of the boundary points between two regions is a good indication of region merging saliency and

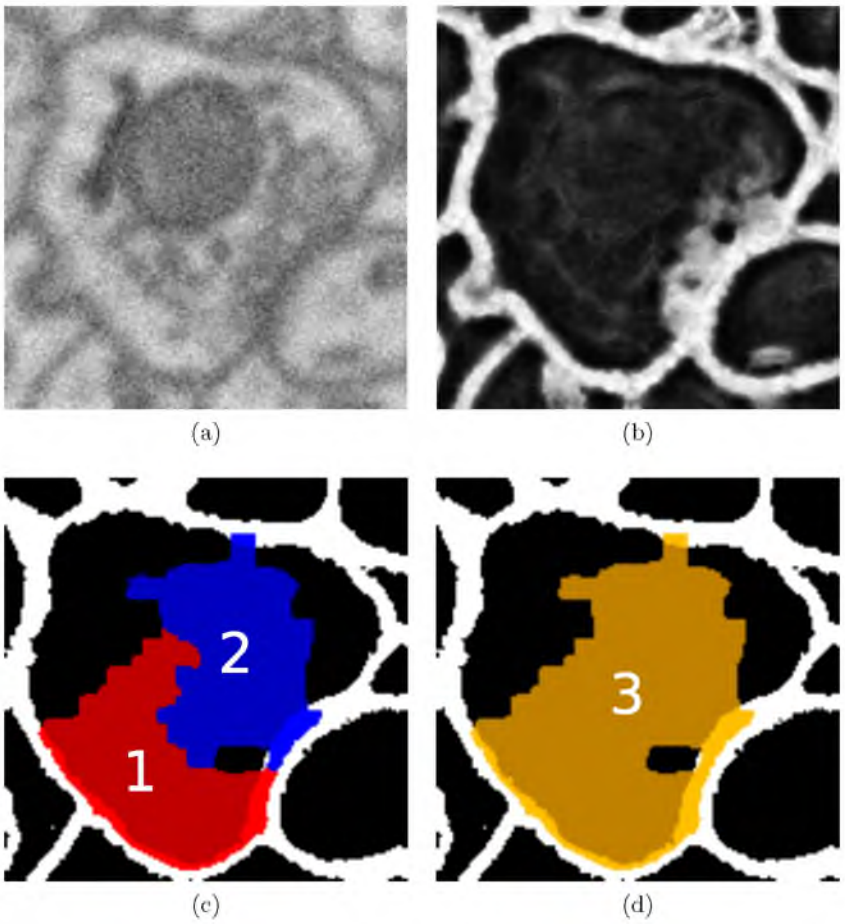


Figure 2.1: Example of (a) an original EM image (zoomed-in), (b) a membrane detection confidence map, (c) two merging regions s_1 (red) and s_2 (blue), and (d) their merging result region s_3 (gold). All regions are overlaid to the ground truth segmentation. The membrane detection confidence map is histogram-equalized for visualization purposes.

gives a more robust boundary measure than other statistics, such as minimum or mean. Thus, the merging saliency function is specified as

$$f_s(s_i, s_j; Pb) = 1 - \text{median}(\{Pb(k) \mid k \in \mathcal{B}(s_i, s_j)\}), \tag{2.3}$$

where $Pb(k)$ is the value of the k-th pixel on a boundary confidence map Pb. We define $f_s(s_i, s_j) = -\infty$ if $\mathcal{B}(s_i, s_j)$ is the empty set.

In practice, some regions in the initial superpixel segmentation can be too small to extract meaningful region-based features (see Section 2.2.3), so we optionally conduct a premerging step to merge initial regions smaller than a certain threshold of $\theta_p^{a_1}$ pixels to one of its neighbor regions that yields the highest merging saliency according to (2.3). We also premerge a region if its size is smaller than a certain threshold of $\theta_p^{a_2}$ pixels, and its

average intensity from the membrane detection probability map of all its pixels is above a certain threshold θ_p^{Pb} as an artifact due to thick membranes.

Consider a graph in which each node corresponds to a superpixel and an edge is defined between two nodes that are neighbors. Starting with the initial oversegmentation S_o, finding a final segmentation, which is essentially the merging of initial superpixels, can be considered as combining nodes and removing edges between them in the graph. This superpixel merging can be done in an iterative fashion: each time a pair of neighboring nodes is combined in the graph with corresponding edges updated, and this process is repeated until only one node is left in the graph. The merging priority can be determined using the merging saliency function f_s. Each time, we combine two nodes whose corresponding regions yield the highest merging saliency,

$$(s_{i^*}, s_{j^*}) = \arg\max_{s_i, s_j \in S} f_s(s_i, s_j; Pb). \tag{2.4}$$

To represent the order of such merging, we use a full binary tree structure, which we call the merge tree. In a merge tree $Tr = (\mathcal{V}, \mathcal{E})$, a node $v_i \in \mathcal{V}$ represents an image region $s_i \in 2^{\mathcal{P}}$. Leaf nodes correspond to initial superpixels in S_o. A nonleaf node corresponds to an image region formed by merging superpixels, and the root node corresponds to the whole image as one single region. An edge $e_{i,c} \in \mathcal{E}$ between node v_i and its child v_c exists if $s_c \subset s_i$, and a local structure $(\{v_i, v_{c_1}, v_{c_2}\}, \{e_{i,c_1}, e_{i,c_2}\})$ represents that region s_i is the merge of region s_{c_1} and s_{c_2}. Figure 2.2c shows a merge tree example with initial superpixels shown in Figure 2.2a. Region s_1 to s_{14} correspond to the leaf nodes v_1 to v_{14} in the merge tree. The nonleaf nodes are the merging result of their descendant regions. For example, region s_{15} is the merge of region s_4 and s_{10}. The root node v_{27} corresponds to the whole image. It is noteworthy that a merge tree defined here can be seen as a dendrogram in hierarchical clustering [14] with each cluster being an image region.

Given the merge tree, finding a segmentation of the image becomes finding a subset of nodes. We call a segmentation formed by a subset of tree nodes a tree-derived segmentation. Among all possible tree-derived segmentations, we call the one that best aligns with the ground truth segmentation the best-effort tree-derived segmentation. Our goal is to select a subset of nodes that form a segmentation that is as close to the best-effort tree-derived segmentation as possible.

2.2.3 Boundary classifier

We use a binary label $y_i \in \{0, 1\}$ to indicate if the region merging at node v_i should occur ("merge", $y_i = 1$) or not ("split", $y_i = 0$). We would like to assign a likelihood score

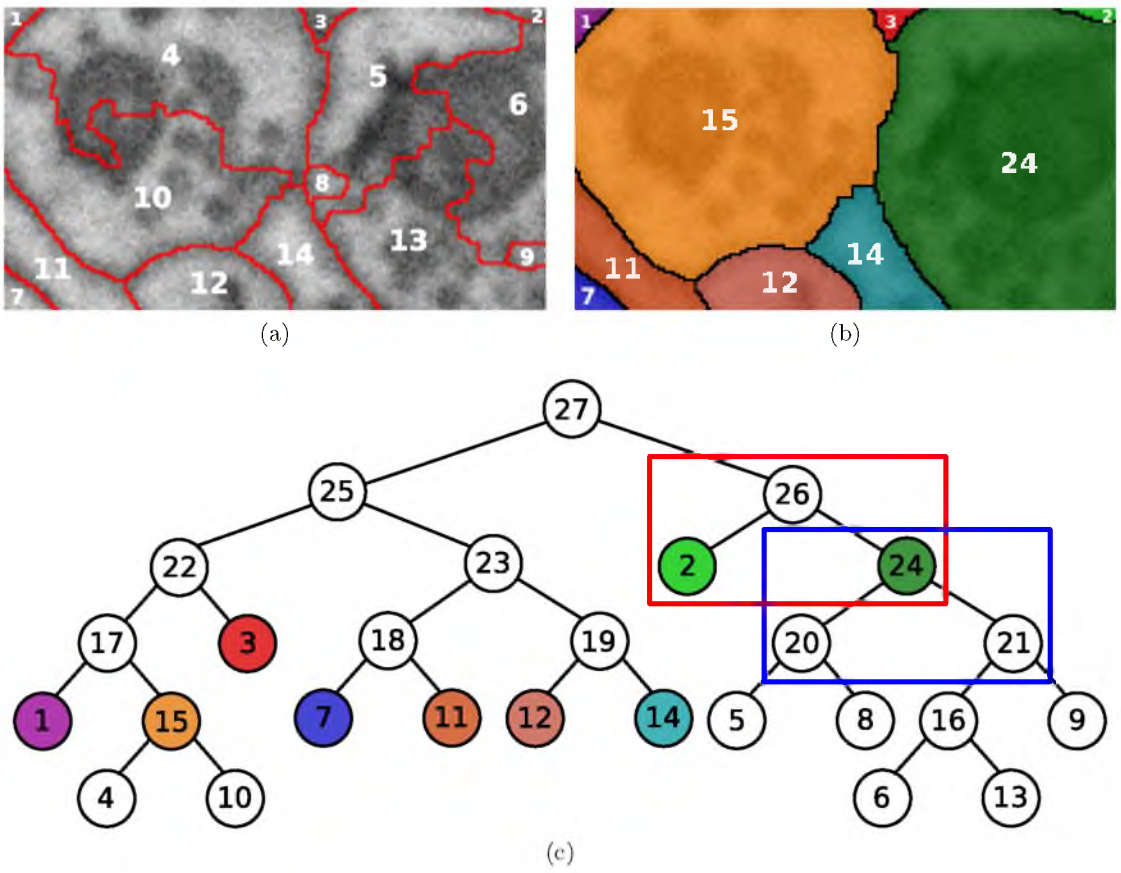


Figure 2.2: Example of (a) an initial segmentation, (b) a consistent final segmentation, both overlaid on the original EM image, and (c) a corresponding merge tree. Each leaf node has the same label as its corresponding initial region, and the colored nodes correspond to regions in the final segmentation. As an example of node potential computation described in Section 2.2.4, the potential of node v_{24} is equal to the probability that node v_{20} and v_{21} merge to node v_{24} (the blue box), while node v_2 and v_{24} do not merge to node v_{26} (the red box).

$P(y_i = 1)$ in order to select the best segments from the merge tree. One possible solution is to use the merging saliency function output. However, it relies only on the boundary cues and cannot utilize the abundant information from the two merging regions. Instead, we use a binary classification function, named boundary classifier. We take advantage of ground truth data and train the boundary classifier with $(\mathbf{X}, \mathbf{y})$, where $\mathbf{X} = \{\mathbf{x}_i\}_i$ is a collection of feature vectors generated at each merge in training images, and $\mathbf{y} = \{y_i\}_i$ is their corresponding labels. The classifier is then used to predict $P(y \,|\, \mathbf{x}) \in [0, 1]$ for any testing case $\mathbf{x}$.

We use nonlocal features computed from each pair of merging regions, including region geometry, image intensity, and texture statistics from both original EM images and

membrane detection probability maps, and merging saliency information. Appendix A.1 summarizes the categories of features used by boundary classifiers. One advantage of our features over the local features commonly used by pixel classifiers [46, 66] is that our features are extracted from regions instead of pixels and thus can be more informative. For instance, we use geometric features to incorporate region shape information for the classification, which is not feasible for pixel classifiers.

We use ground truth segmentations to generate true merge labels $\mathbf{y}$ for training. We compare the errors under a certain metric for both merging (ε_m) and keeping split (ε_s) against the ground truth. Either case with smaller error deviates less from the ground truth and should thus be adopted. In the example shown in Figure 2.1, region s_3 aligns better with the ground truth, so region s_1 and s_2 should be combined. The training label is determined automatically as

$$y_k = \begin{cases} 1 & \text{if } \varepsilon_m < \varepsilon_s \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

The error metric can be chosen to reflect favor of certain segmentation quality. We use the adapted Rand F-error (Section 2.3.2) for EM image segmentation.

The boundary classifier is not limited to a specific classification model. We use the random forest [67] and the logistic disjunctive normal network in practice. For random forest, we assign different weights to positive ($y = 1$) and negative ($y = 0$) training samples to balance their contributions. The weights w^1_{rf} for positive samples and w^0_{rf} for negative samples are determined as

$$w^1_{\text{rf}} = \begin{cases} 1 & \text{if } N^1_{\text{rf}} > N^0_{\text{rf}} \\ N^0_{\text{rf}} / N^1_{\text{rf}} & \text{otherwise,} \end{cases} \tag{2.6}$$

$$w^0_{\text{rf}} = \begin{cases} N^1_{\text{rf}} / N^0_{\text{rf}} & \text{if } N^1_{\text{rf}} > N^0_{\text{rf}} \\ 1 & \text{otherwise,} \end{cases} \tag{2.7}$$

where N^1_{rf} and N^0_{rf} are the number of samples with $y = 0$ and $y = 1$, respectively.

2.2.4 Inference

After creating the merge tree, the task of generating a final segmentation of the image becomes choosing a subset of nodes in the merge tree. It is equivalent to finding a complete binary label assignment $\mathbf{z} = \{z_i\}_{i=1}^{|\mathcal{V}|}$ for every node being a final segment ($z = 1$) or not ($z = 0$). The labeling must preserve the pixel consistency of the final segmentation. The pixel consistency requires that any pixel should be labeled exactly once in the final segmentation for belonging to one unique connected component. In a merge tree, the pixel

consistency requires that if a node is selected, all of its ancestors or descendants cannot be selected; if a node is not selected, one of its ancestors or a set of its descendants must be selected. In other words, exactly one node should be selected on any path from a leaf node to the root node. Figure 2.2c shows an example: the colored nodes are picked to form a consistent final segmentation in Figure 2.2b. The other nodes cannot be picked, because we cannot label any pixel more than once. For example, if node v_4 (or v_{10}) is also picked along with node v_{15}, the pixels in region s_4 (s_{10}) would be labeled as both 15 and 4 (or 10), which violates the pixel consistency by definition.

Let $\rho(i)$ be a query function that returns the index of the parent of node v_i. The k-th ($k = 1, \dots d_i$) ancestor of v_i is denoted as $\rho^k(i)$ with d_i being the depth of v_i in the tree, and $\rho^0(i) = i$. We refer to the constraint mentioned above as the *region consistency constraint*:

$$\sum_{k=0}^{d_i} z_{\rho^k(i)} = 1, \tag{2.8}$$

which we enforce for every leaf-to-root path starting at node v_i.

Based on the boundary classifier predictions, we assign a potential to each node as the likelihood that the node represents a best-effort segment. Considering 1) a region exists in the final segmentation because it neither splits into smaller regions nor merges into others, and 2) the Markov assumption that each prediction the classifier makes depends only on the local merge structure, we define the potential for a node v_i as

$$U_i = P(y_i = 1) \cdot P(y_{\rho(i)} = 0). \tag{2.9}$$

Intuitively, the node potential U_i is the probability that the children of node v_i should be combined, but node v_i should not be combined with its sibling into its parent $v_{\rho(i)}$. As an example shown in Figure 2.2c, the necessary condition for region s_{24} being a final segment is that the merge below it (the blue box) happens and the merge above it does not (the red box), so the potential of node v_{24} is $U_{24} = P(y_{24} = 1) \cdot P(y_{26} = 0)$. Since a leaf node v_i has no children, its potential is computed as $P(y_{\rho(i)} = 0)^2$. Similarly, the potential for the root node v_r is computed as $P(y_r = 1)^2$.

Under the region consistency constraint, we apply a greedy optimization algorithm to infer the label assignments $\mathbf{z}$. First, we label an unlabeled node that has the highest potential in the merge tree with $z = 1$. Then, the ancestors and descendants of this node are labeled $z = 0$ as inconsistent choices. This procedure is repeated until every node in the merge tree is labeled. The set of nodes with $z = 1$ forms a consistent final segmentation.

2.3 Results

We validate our proposed method using three EM image data sets.

2.3.1 Data sets

2.3.1.1 *Drosophila* VNC data set

The *Drosophila melanogaster* first instar larval ventral nerve cord (VNC) data set [37, 38] contains 60 512×512 images acquired using SSTEM at the resolution of $4 \times 4 \times 50$ nm/pixel. This data set was used in ISBI 2012 EM Segmentation Challenge [68] with ground truth 2D segmentations of 30 consecutive images as the training set and the other 30 consecutive images as the testing set. We target 2D segmentation for this data set. We train our algorithm with the 30 training images, test on the 30 testing images, and submit the results to the challenge website for evaluation due to the unavailability of ground truth for the testing images.

2.3.1.2 Mouse neuropil data set

The whole mouse neuropil data set [69] is a stack of 400 images of size 4096×4096 acquired using SBFSEM. The resolution is $10 \times 10 \times 50$ nm/pixel. A subset of 70 700×700 images is cropped and the 2D ground truth segmentations are annotated by a human expert for performance evaluation. We target 2D segmentation for this data set. A subset of 14 images is randomly selected to train our algorithm, and the remaining 56 images are used for testing.

2.3.1.3 Mouse cortex data set

Also known as the AC4 data set, the whole mouse cortex data set is a stack of 1850 images of 4096×4096 pixels acquired by SSSEM at the resolution of $3 \times 3 \times 30$ nm/pixel. The images were down-sampled by a factor of 2 in the x-y plane, resulting in $6 \times 6 \times 30$ nm/pixel resolution. Two subsets of $1024 \times 1024 \times 100$ pixels were cropped and used in ISBI 2013 SNEMI3D Challenge [70] as the training and the testing sets, respectively. We target 3D segmentation. We train our algorithm on the training stack, test on the testing stack, and submit the results to the challenge website for evaluation due to the unavailability of ground truth for the testing images.

2.3.2 Evaluation metrics

We use the adapted Rand F-error [39] as the metric for both training label generation and result evaluation. Unlike traditional pixel classification error metrics, the Rand F-error is

sensitive to incorrect region separation but less sensitive to minor shifts of region boundaries. It is used as the standard error metric by the public EM segmentation challenges [68, 70].

Similar to Rand index [71], the Rand F-error is based on pairwise pixel metric that examines the labels of every pair of pixels from the proposed segmentation S and the ground truth segmentation S_g and classifies it as one of the four categories: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Specifically, the number of pixel pairs in each category is computed as

$$N_{\mathrm{TP}} = \sum_{i<j} \mathbb{I}(S(i) = S(j) \wedge S_g(i) = S_g(j)), \tag{2.10}$$

$$N_{\mathrm{TN}} = \sum_{i<j} \mathbb{I}(S(i) \neq S(j) \wedge S_g(i) \neq S_g(j)), \tag{2.11}$$

$$N_{\mathrm{FP}} = \sum_{i<j} \mathbb{I}(S(i) = S(j) \wedge S_g(i) \neq S_g(j)), \tag{2.12}$$

$$N_{\mathrm{FN}} = \sum_{i<j} \mathbb{I}(S(i) \neq S(j) \wedge S_g(i) = S_g(j)), \tag{2.13}$$

where $S(i)$ is the label of the i-th pixel in S, and $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the input proposition is true or 0 otherwise. Intuitively, TP and TN pixel pairs are those that are correctly combined or separated in the proposed segmentation. FP pixel pairs are those that are incorrectly combined and lead to undersegmentation error. FN pixel pairs are those that are incorrectly separated and lead to oversegmentation error.

With precision and recall defined as

$$\text{Precision} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{FP}}}, \tag{2.14}$$

$$\text{Recall} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{FN}}}, \tag{2.15}$$

the Rand F-error is computed as

$$\text{Rand F-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{2.16}$$

$$\text{Rand F-error} = 1 - \text{Rand F-score}. \tag{2.17}$$

The values of precision, recall, Rand F-score, and the Rand F-error are all in the range between 0 and 1. High precision indicates low undersegmentation error, and high recall indicates low oversegmentation error. Zero Rand F-error indicates a perfect segmentation.

Since minor shifts of region boundary pixels are not important in the applications of EM segmentation, the actual Rand F-error we use is adapted to tolerate such errors by ignoring pixels that have the membrane (background) label in ground truth segmentations. We refer to the Rand F-error evaluated in this adaptive way as the adapted Rand F-error.

2.3.3 Experiments

2.3.3.1 *Drosophila* VNC data set

For the *Drosophila* VNC data set, we use both the membrane detection probability maps generated using the Cascaded Hierarchical Model (CHM) [66] and deep convolutional neural networks (DNN) [46] trained with 30 training images. The watershed local minimum dynamic threshold θ_w is used as 0.03 and 0.01 for the CHM and the DNN probability maps, respectively. The precisions and recalls of the initial segmentations of training images are shown in Table 2.1. The high precisions and relatively low recalls indicate we ensured initial oversegmentation. For both data sets and probability maps, we apply the premerging step with parameters $\theta_p^{a_1} = 50$, $\theta_p^{a_2} = 200$ and $\theta_p^{Pb} = 0.5$. For the random forest used as our boundary classifier, we use $\theta_{\text{rf}}^t = 255$ trees with $\theta_{\text{rf}}^s = 70\%$ of all training samples randomly selected to train each tree, and at each decision tree node, the square root of the number of features is examined for the most informative one for branching.

The testing results are shown in Table 2.2 along with the results of other state-of-the-art methods from various groups for comparison. All the results are also available on the ISBI 2012 EM Segmentation Challenge online leader board [68], which currently still accepts submissions for evaluation after the challenge. Note that the challenge evaluation system thresholds the resulting images at 11 thresholds uniformly distributed between 0 and 1 and selects the best result. Therefore, the resulting images as probability maps are thresholded at the best thresholds, whereas other resulting images as hard segmentations, such as ours, yield identical results with different thresholds. The "Human" entries are generated by two individual human observers.

In Table 2.2, entries with group name "IDSIA" use the DNN membrane detection [46] probability maps; Entry 7 is the result by applying HMT to the CHM membrane detection [66] probability maps; Entries 3, 5, and 9 are from multigroup collaborations, in which the first groups provide membrane detection results and the latter groups apply region-based segmentation methods afterwards. We can see that applying our approach consistently improves the adapted Rand F-error from membrane detection results (comparing Entry 11

Table 2.1: Precisions and recalls of initial segmentations of the *Drosophila* VNC data set training images using different probability maps. The watershed threshold θ_w is 0.03 for the CHM and 0.01 for the DNN probability maps.

Method	Precision	Recall
CHM [46]	0.9967	0.5054
DNN [66]	0.9971	0.9704

Table 2.2: 2D segmentation results (adapted Rand F-error) of the *Drosophila* VNC data set. "Human 1" and "Human 2" represent manually labeled results by two human observers. The results are also available on the ISBI 2012 EM Challenge online leader board. Our results are under the group name "SCI." The "+" in group names indicate multigroup collaborations.

Rank	Group	Adapted Rand F-error
-	Human 1	0.002109
1	Heidelberg/HCI	0.01155
2	CUHK	0.02318
3	ODU + **SCI**	**0.02377**
4	JHU/APL	0.02522
5	IDSIA + **SCI**	**0.02698**
6	Freiburg	0.02724
7	**SCI**	**0.02856**
8	"Masters"	0.02989
-	Human 2	0.02995
9	IDSIA + Rutgers	0.03010
10	IDSIA	0.03101
11	ODU	0.03143
12	IDSIA (LSTM)	0.03233
13	"Connectome"	0.03799
14	ETH/INI	0.03910
15	MIT	0.03928
-	Thresholding	0.2755

vs. Entry 3 and Entry 10 vs. Entry 5). Using either probability maps, our approach yields even smaller errors than a human observer ("Human 2"). Based on the results, we claim that HMT can improve 2D segmentation accuracy from thresholding membrane probability maps with the best thresholds independent of the membrane detection algorithms. It is noteworthy that Entry "IDSIA-SCI" led the challenge since 2013 until the emergence of recent submissions, especially the current leader "Heidelberg/HCI," which made the most significant improvement seen in years.

Figure 2.3 shows four example testing results of both pixelwise membrane detections and HMT 2D segmentation. Our approach closes boundary gaps in the membrane detection, which may lead to undersegmentation errors by thresholding. Our approach also removes some undesirable intracellular structures.

2.3.3.2 Mouse neuropil data set

For the mouse neuropil data set, since the membrane detection results using DNN [46] are not available, we experiment only with the probability maps generated by CHM [66].

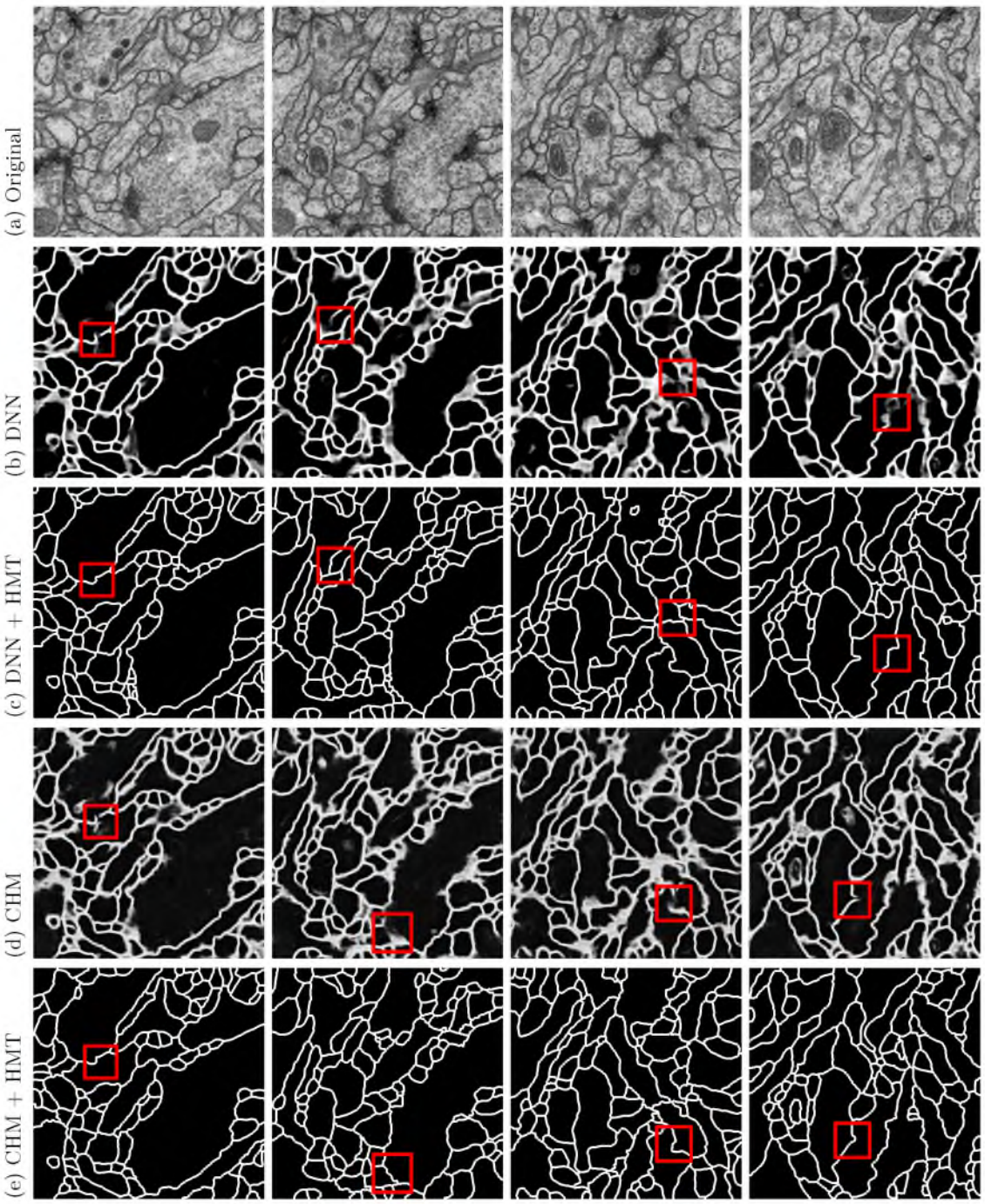


Figure 2.3: Examples of 2D segmentation results (four sections) of the *Drosophila* VNC data set, including (in rows) (a) original EM images, (b) DNN membrane detection, (c) HMT segmentation using the DNN results, (d) CHM membrane detection, and (e) HMT segmentation using the CHM results. The resulting cell boundaries are thickened for visualization purposes. The red boxes show examples of missing boundaries fixed by HMT.

The membrane detection classifier is trained with 14 images that are randomly selected. We train our algorithm with the same 14 images and test on the rest of the stack. We use $\theta_w = 0.02$ to threshold the watershed local minima, and the same parameters for the premerging ($\theta_p^{a_1} = 50$, $\theta_p^{a_2} = 200$ and $\theta_p^{Pb} = 0.5$) and the random forest ($\theta_{\text{rf}}^t = 255$, $\theta_{\text{rf}}^s = 0.7$) as in the *Drosophila* VNC experiments. The adapted Rand F-errors of the training and testing set are shown in Table 2.3, which we also compare with thresholding the membrane detection probability maps at the best value.

In general, the mouse neuropil data set is a more difficult data set, because of its larger variations of cell shapes and more complex intracellular structures. HMT again has a significant 0.073 improvement over thresholding the membrane detection probabilities. Figure 2.4 shows four example testing images of HMT compared with membrane probability maps. We can see that our method is able to close gaps on the membranes detected by the pixelwise algorithm.

2.3.3.3 Mouse cortex data set

For the mouse cortex data set, we use the membrane detection probability maps generated using DNN [46] trained with 100 2D images. We use 3D watershed transform over stacks of 2D probability maps with $\theta_w = 0.005$ to generate 3D superpixels, which we use HMT to combine for final segmentations. The premerging parameters are $\theta_p^{a_1} = 500$, $\theta_p^{a_2} = 3000$, and $\theta_p^{Pb} = 0.5$. For the boundary classifier, we use the logistic disjunctive normal network [72] with $\theta_{\text{LDNN}}^{n_g} = 10$ groups and $\theta_{\text{LDNN}}^{n_d} = 10$ discriminants per group. The testing results are submitted to the ISBI 2013 SNEMI3D Challenge [70] website for evaluation. The adapted Rand F-errors are computed for the whole 3D testing stack. Table 2.4 shows the top entries on the current challenge leader board. Visualization of selected testing 3D neuron segmentation is generated using TrakEM2 [38] in Fiji [73] and shown in Figure 2.5.

When developed, our algorithm was one of the state-of-the-art methods until the very recent emergence of the top entry under group name "Heidelberg/HCI." It is noteworthy that applying HMT to directly merge 3D superpixels (Entry 3) outperforms using a two-step

Table 2.3: 2D segmentation results (adapted Rand F-error) of the mouse neuropil data set. "Thresholding" refers to the region segmentation result by labeling connected components from thresholded membrane detection confidence maps at a globally best value.

Method	Training	Testing
Thresholding	0.07298	0.2023
HMT	**0.04128**	**0.1288**

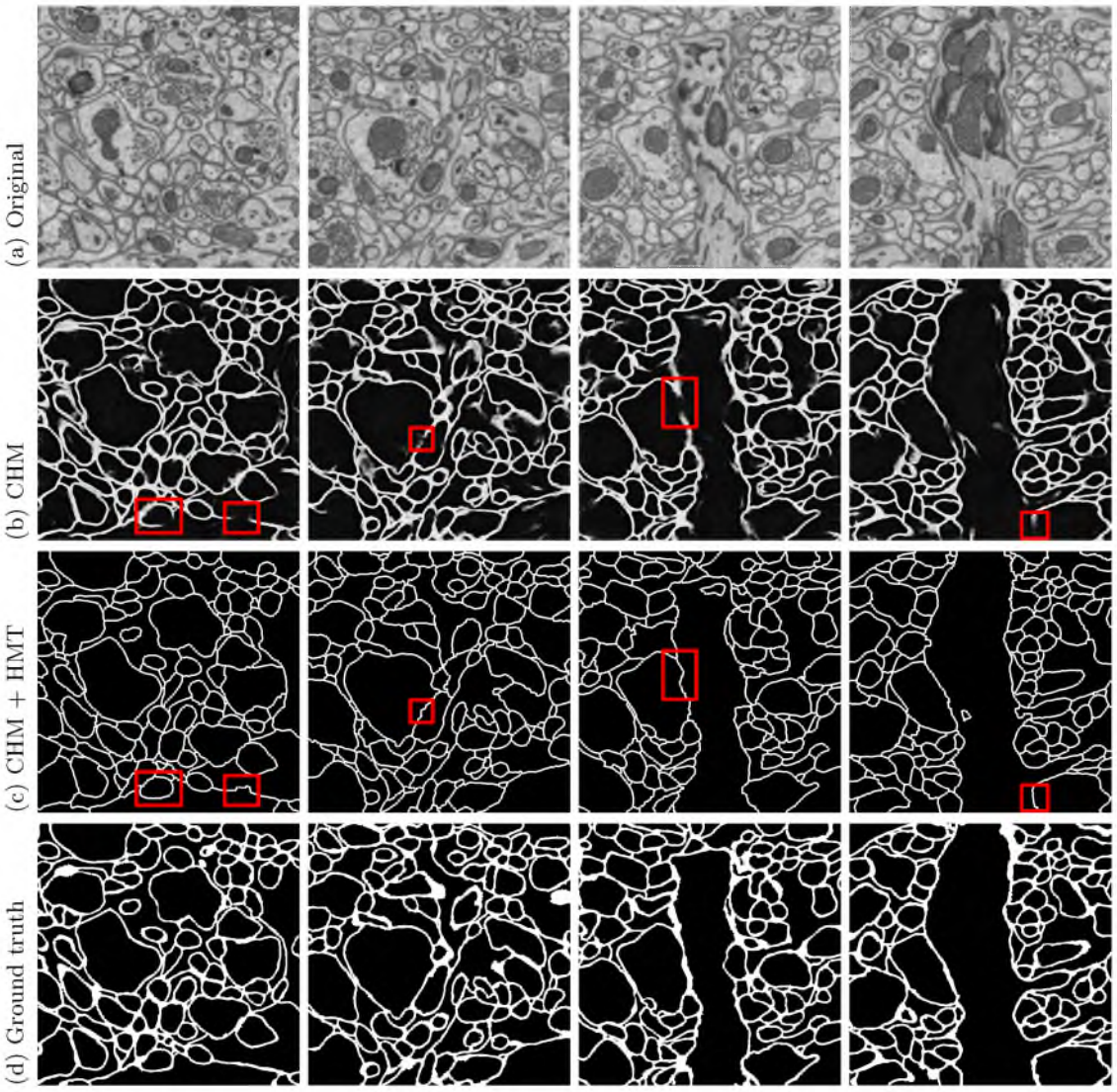


Figure 2.4: Examples of 2D segmentation results (four sections) of the mouse neuropil data set, including (in row): (a) original EM images, (b) CHM membrane detection, (c) HMT segmentation results, and (d) ground truth images. The resulting cell boundaries are thickened for visualization purposes. The red boxes show examples of missing boundaries fixed by HMT.

Table 2.4: 3D segmentation results (adapted Rand F-error) of the mouse cortex data set. "Human" represents manually labeled results by a human observer. The results are also available on the ISBI 2013 SNEMI3D Challenge online leader board. Our results are under the group name "SCI." The "+" in group names indicates multigroup collaborations.

Rank	Group	Adapted Rand F-error
-	Human	0.05998
1	Heidelberg/HCI	0.07432
2	Janelia Farm + Princeton	0.1004
3	**SCI**	**0.1083**
4	MIT	0.1136
5	ODU	0.1146
6	Janelia Farm	0.1250
7	SCI (2D HMT + 3D Linking)	0.1315
8	Harvard	0.1484
9	Singapore ASTAR	0.1665
10	Heidelberg/MPI	0.2044

approach [41] that uses HMT to segment each individual 2D section and identifies inter-section links using a section classifier (Entry 7). This is because 2D segmentation errors can be propagated to affect inter-section linking in the two-step approach. Also, the section classifier can predict only the similarity between a pair of regions and cannot handle the cases of neuron merging or branching. On the contrary, the boundary classifier for 3D HMT is trained with 3D features and is better at characterizing spatial region combinations. In addition, since HMT makes no assumption about region shapes and topology, the merging and branching of neurons are handled naturally in the HMT framework.

2.4 Conclusion

We developed a fully automatic approach to hierarchical segmentation of EM images. According to the experimental results, the HMT model improves neuron segmentation accuracy substantially compared with thresholding the membrane probability maps at all levels. By using superpixels instead of pixels as the unit element, we are able to compute richer region-based features. Also, the use of the merge tree structure presents the most plausible segmentation hypotheses in a more efficient way than using a general graph structure, and it transforms the problem of final segmentation inference from considering all possible superpixel combinations to choosing from a set of given answers. The way the node potentials are evaluated incorporates information from both lower and higher merging levels, and thus the impact of single boundary classification error can be alleviated. As we can see so far, one major concern about using the automated algorithm based on the

Figure 2.5: Examples of 3D neuron segmentation results of the mouse cortex data set. Different colors indicate individual segments.

merge tree structure is its inability to fix incorrect region merging orders. According to the experimental results, however, we argue that boundary median probability is a robust merging saliency metric, which helps generate the correct merging order for most cases. Also, with further improvement of membrane detection algorithms, we will have more consistent membrane probability maps as input, and occurrences of incorrect merging orders that actually lead to incorrect segmentation will be further suppressed.

CHAPTER 3

HIERARCHICAL MERGE FOREST MODEL

When we segment 2D EM images using HMT (Chapter 2), one missing factor is the inter-section information. Despite the anisotropy of image volumes, substantial region similarities can be observed across consecutive sections. Corresponding regions on adjacent sections may provide important clues about segmenting a current section. This is in fact what human experts do when a 2D segmentation decision is difficult to make: adjacent sections are looked at for corresponding regions that belong to the same neuron, and the geometric and/or textural information from such corresponding regions is used for assistance. In this chapter, we propose a simulation to this procedure, which is also presented in [74]. We extend the HMT model by introducing a section classifier to identify region correspondence between adjacent sections. Then, all trees, each representing one section, are combined with their node potentials updated according to inter-section correspondences. Instead of resolving one single tree at a time, we infer the final segmentation of each section simultaneously from jointly from all trees. In this way, we take advantage of inter-section information and improve the overall 2D segmentation accuracy as demonstrated by the experimental results on two EM image data sets.

3.1 Introduction

Most current EM modalities generate anisotropic image volumes. The lower vertical resolution makes direct 3D neuron segmentation difficult, and two-step approaches that first segment profiles of neuron cells in each 2D image section and then link them across sections to reconstruction 3D neuron models are widely used [24, 53, 54, 41]. In these methods, 2D segmentation error can be propagated during the linking procedure and deteriorate 3D reconstruction quality. Therefore, improving 2D segmentation accuracy is our focus in this chapter. Because of the high resolutions of EM images, most neurons appear in more than one image sections. Despite the potential variations in appearance between pairs of

2D profiles of the same 3D neuron, abundant information usually exists that helps human experts identify the correspondences of such profile pairs. For instance, two consecutive image sections of the *Drosophila* VNC data set (Section 2.3.1.1) are shown in Figure 3.1. In spite of the $4 \times 4 \times 50$ nm/pixel anisotropic resolution and considerable coordinate shift between the two sections, we can still clearly see the cell correspondences indicated by visual similarity. Such correspondences are usually used in turn to fix 2D annotation errors in manual analysis as well as semi-automatic and fully automatic neuron segmentation methods [24, 52, 57, 58, 41, 75]. For example, segmenting the image area in the red box in Figure 3.1a is difficult. However, it can be resolved by looking at the same location in the next section (Figure 3.1b). In order to use such correspondences in fully automatic algorithms, we need algorithms that are able to detect region correspondences reliably under complex situations, such as considerable image deformation, misregistration, quality difference, etc.

In this chapter, we extend our HMT model and propose a fully automatic method to utilize inter-section information for neuron segmentation in EM 2D image stacks, which we call the hierarchical merge forest (HMF) model. We build a merge forest structure by combining merge trees that represent the region merging hierarchy of each 2D section in the stack. A section classifier is learned to identify the most likely region correspondences between adjacent sections. The inter-section information from such correspondences is

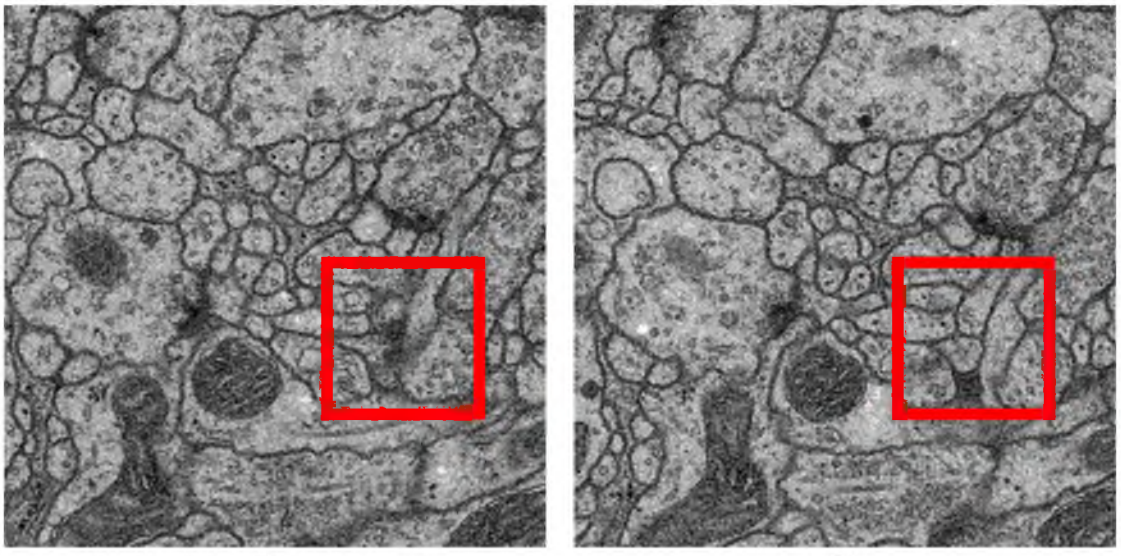


Figure 3.1: Examples of two consecutive 2D image sections of the *Drosophila* VNC data set. The red boxes are placed at the same location in the two sections.

incorporated to update the potentials of tree nodes. We resolve the merge forest using these potentials together with consistency constraints to acquire the final segmentation of each 2D section in the stack. We demonstrate that our method leads to segmentation accuracy improvement by experimenting with two types of EM image data sets.

Most of the works on segmenting EM images that explicitly use inter-section information are for simultaneous 2D segmentation and 3D reconstruction [52, 57, 58]. The most related work is proposed by Funke et al. [58]. They use a similar tree structure to represent the organization of segmentation hypotheses for each image and combine them into a forest. Their approach considers features only from regions located in different sections but not from regions within a section. In addition, it optimizes only inter-section connections to acquire 2D segmentation with no respect to 2D segmentation quality. On the contrary, our framework utilizes both nonlocal image features from individual 2D sections for learning the boundary classifier and similarity features between spatially adjacent region pairs in different sections for learning the section classifier. We then combine intra-section region merging and inter-section region correspondence likelihood predictions from the two classifiers to infer 2D segmentation of each section in the stack.

It is noteworthy that a group of methods called cosegmentation in the computer vision community jointly segment similar objects in a set of images by utilizing the shared features between objects in the same class [76, 77, 78]. These methods usually need object-dependent semantic information, such as specific shapes and texture appearances, about the objects that are being segmented. Also, they usually work only for one-to-one correspondence and do not handle multicorrespondence or uncorresponded objects. Therefore, they are not directly suitable for neuron segmentation in EM images, for which it is common that 3D merging/branching of neurons cause merges/splits of their 2D profiles, and neuron terminals result in missing correspondence between adjacent 2D sections.

3.2 Methodology

3.2.1 Merge forest

Since the region merging hierarchy of one section can be represented by a merge tree, it is straightforward that we can use a series of merge trees, or in other words a merge forest, to represent an image stack consisting of consecutive sections. To compute how probable a node should be in the final segmentation, we not only consider the node potential from the boundary classifier, but also refer to corresponding nodes in adjacent sections, which we call reference nodes. Connections to all possible reference nodes can be considered as

directed edges between nodes in different trees, which we call reference edges. The most likely corresponding node is called the best reference node and the corresponding edge is called the best reference edge (explained in detail in Section 3.2.3). We use $v_{(m,i)}$ to denote the i-th node in section m. Figure 3.2 shows an example: node $v_{(m,1)}$ has node $v_{(m-1,2)}$, $v_{(m-1,3)}$, $v_{(m-1,4)}$, $v_{(m+1,5)}$, $v_{(m+1,6)}$, and $v_{(m+1,7)}$ as possible reference nodes, and thus there are reference edges from node $v_{(m,1)}$ to these nodes. Suppose node $v_{(m-1,3)}$ is the best reference node of node $v_{(m,1)}$, then the edge from node $v_{(m,1)}$ to $v_{(m-1,3)}$ is the best reference edge. We denote a reference edge from node $v_{(m,i)}$ in section m to its reference node $v_{(m_r,i_r)}$ in an adjacent section m_r as $e_{(m,i),(m_r,i_r)}$.

Regions that are too large or too far away are eliminated as very unlikely reference node choices, and therefore, the number of reference nodes/edges is linearly proportional to the number of nodes in a forest. Due to the anisotropic nature of most EM image data sets with which we experiment, we use $m_r = m \pm 1$ in practice.

3.2.2 Section classifier

Reference edges do not always represent true region correspondences. On one hand, two consecutive 2D profiles of the same 3D neuron usually have relatable looks, so the correspondence between two similar-looking regions is more likely to be true. On the other hand, the veracity of a reference edge should not be determined merely based on the similarity between the region pair, because there can be cases in which the two regions are not best-effort segments but have a very similar appearance. Uplifting the likelihood of their correspondence may lead to confusion. Therefore, we define a reference edge is true

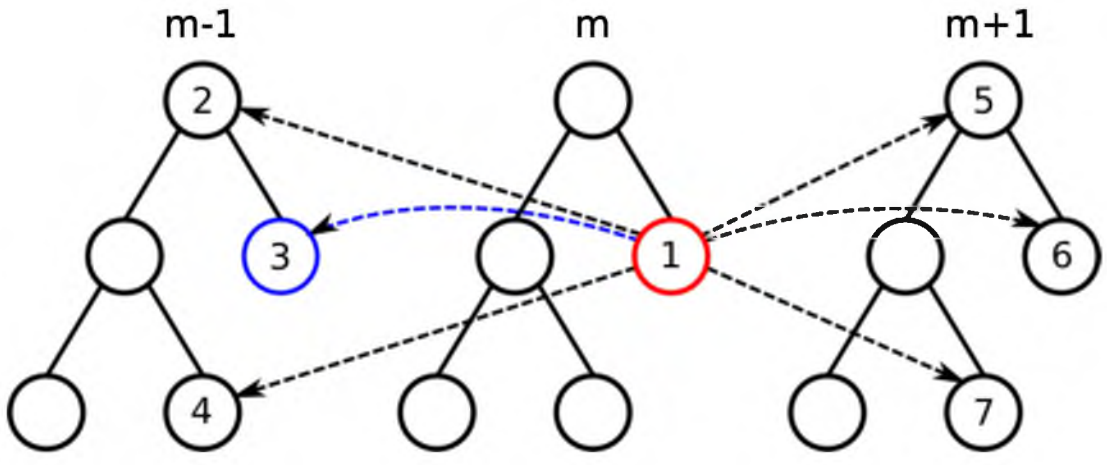


Figure 3.2: Example of a merge forest consisting of three merge trees with reference edges. The arrows are reference edges of node $v_{(m,1)}$, and their end nodes are its reference nodes. Node $v_{(m-1,3)}$ is the best reference node of $v_{(m,1)}$.

only if both its end nodes belong to the same 3D object and also correspond to best-effort image segments. To identify true correspondences, we introduce a section classifier. Let $u_{(m,i),(m_r,i_r)} \in \{0,1\}$ denote whether a reference edge $e_{(m,i),(m_r,i_r)}$ is true ($u_{(m,i),(m_r,i_r)} = 1$) or not ($u_{(m,i),(m_r,i_r)} = 0$). The section classifier is designed to predict $P(u_{(m,i),(m_r,i_r)} = 1)$.

The section classifier takes a set of features computed from a pair of potentially corresponded regions, including geometric features (region area/perimeter/compactness differences, centroid distance, overlapping, etc.), image intensity statistics features (region and boundary pixel intensity statistics from both Gaussian denoised EM images and membrane detection maps), and textural features (texton statistics). Appendix A.2 summarizes the categories of features used by section classifiers. We train a random forest classifier [67] with class weights reversely proportional to numbers of positive/negative examples to handle the data imbalance.

3.2.3 Inference

The section classifier predicts how likely each reference edge is true, based on which we choose the best correspondence to update our knowledge about a current node. First, we identify the best reference node based on the section classifier output edge veracity likelihood for each reference edge as

$$(m_r^*, i_r^*) = \arg\max_{(m_r,i_r)} P(u_{(m,i),(m_r,i_r)} = 1). \tag{3.1}$$

Then, each node potential is updated with the likelihood score of its best reference edge and its best reference node potential as

$$V_{(m,i)} = U_{(m,i)} \cdot P(u_{(m,i),(m_r^*,i_r^*)} = 1) \cdot U_{(m_r^*,i_r^*)}, \tag{3.2}$$

where $U_{(m,i)}$ and $U_{(m_r^*,i_r^*)}$ are computed using (2.9) within each tree. With (3.2), we associate the potential of this node with its best reference node, and therefore correlate the chances of existence of the two nodes in the final segmentation. Zero edge weights are set to a minimal positive value ϵ_e that is smaller than the minimum nonzero edge weight overall, because otherwise, the corresponding node potentials would all be punished to exactly zero. If a node has no reference edge, its potential is updated by multiplying ϵ_e and 0.25 as a reference node potential that represents random merge/split decisions from the boundary classifier according to (3.2). In practice, we use $\epsilon_e = 10^{-4}$.

The next step is to resolve the merge forest, which selects a subset of nodes from each tree. The region consistency constraint (Section 2.2.4) still applies: any pixel should be labeled only once. Therefore, if a node is selected, its ancestors and descendants must be

removed. Instead of resolving each merge tree independently, we resolve the whole merge forest simultaneously using a greedy approach. The node with the highest potential in the forest is picked. Then, all of its ancestors and descendants within the tree, which are inconsistent choices, are removed, and all reference edges directed to these removed inconsistent nodes are removed as well. Next, all node potentials are recomputed, and this procedure is repeated until no nodes are left in the forest. The set of selected nodes in each tree forms a consistent final segmentation for all sections.

3.3 Results

We validate our methods using two data sets. One is the mouse neuropil data set (Section 2.3.1.2). Since HMF needs to perform on consecutive sections, we use a different setting from Section 2.3.3.2 in which the last 25 sections are used for training, and the first 45 sections are for testing. The other one is the training stack of the *Drosophila* VNC data set (Section 2.3.1.1). We also use a different setting from Section 2.3.3.1 in which the first 20 sections are used for training and the rest for testing. The ground truth 2D intra-section segmentation and 3D inter-section region correspondence are manually annotated. The hypothetical regions from either the initial segmentation or the region merging are matched to the 2D ground truth regions with respect to symmetric difference in order to generate the training labels for the section classifier.

We use a random forest implementation [79]. The pixelwise membrane detection random forest uses 200 trees. The initial watershed threshold is $\theta_w = 0.01$ and 0.05 for the mouse neuropil and the *Drosophila* VNC data set, respectively. Reference edges are considered between regions smaller than $\theta^a_{\text{mf}} = 200000$ and 40000 pixels and within a centroid distance of $\theta^d_{\text{mf}} = 45$ and 30 pixels for the mouse neuropil and the *Drosophila* VNC data set, respectively, based on their different resolutions and cell sizes. For both the boundary and the section classifier, we use $\theta^t_{\text{rf}} = 255$ trees and $\theta^s_{\text{rf}} = 70\%$ of all samples to train each tree.

We use the adapted Rand F-error [39] as the evaluation error metric (Section 2.3.2). The results for the two data sets via thresholding pixelwise membrane detection results at best threshold, HMT (Chapter 2), and the HMF model in this paper are shown in Table 3.1 for comparison. Figure 3.3 shows zoomed-in examples of the testing images from both data sets.

We can see from Figure 3.3 that HMF can correct node selection mistakes in a section by utilizing information from adjacent sections that are easier to segment. Therefore, it can fix both the oversegmentation and the undersegmentation errors from HMT. The testing

Table 3.1: 2D segmentation results (adapted Rand F-error) of the mouse neuropil and the *Drosophila* VNC data sets by optimally thresholding membrane detection probability maps, HMT and HMF.

(a) Mouse neuropil

Method	Training	Testing
Thresholding	0.1942	0.2946
HMT	0.06768	0.2010
HMF	**0.04917**	**0.1632**

(b) *Drosophila* VNC

Method	Training	Testing
Thresholding	0.1542	0.2449
HMT	0.02656	0.1173
HMF	**0.01937**	**0.08129**

results of the two data sets are improved significantly by over 0.0360 compared with the merge tree results and over 0.131 compared with the thresholding results. Considering this method is meant to be applied to large-scale data sets, the improvement may save a substantial amount of manual work for biologists and neuroscientists.

3.4 Conclusion

In this chapter, we presented an effective extension to our previous HMT model that utilizes inter-section information to improve intra-section neuron segmentation accuracy. In addition to cell continuation as the major type of region connection, we argue that our reference model works for most cell terminations and branchings as well. Since cells appear almost always in more than one section, even if a cell terminates in the next section, correspondence should still be found in the previous section. Also, when a cell branches, its profile often splits unevenly to a similarly sized region and some other much smaller regions, so we expect to find informative reference nodes for most branching cases as well. In future work, we will introduce new features to further improve the section classifier and address the inter-section neuron reconstruction problem.

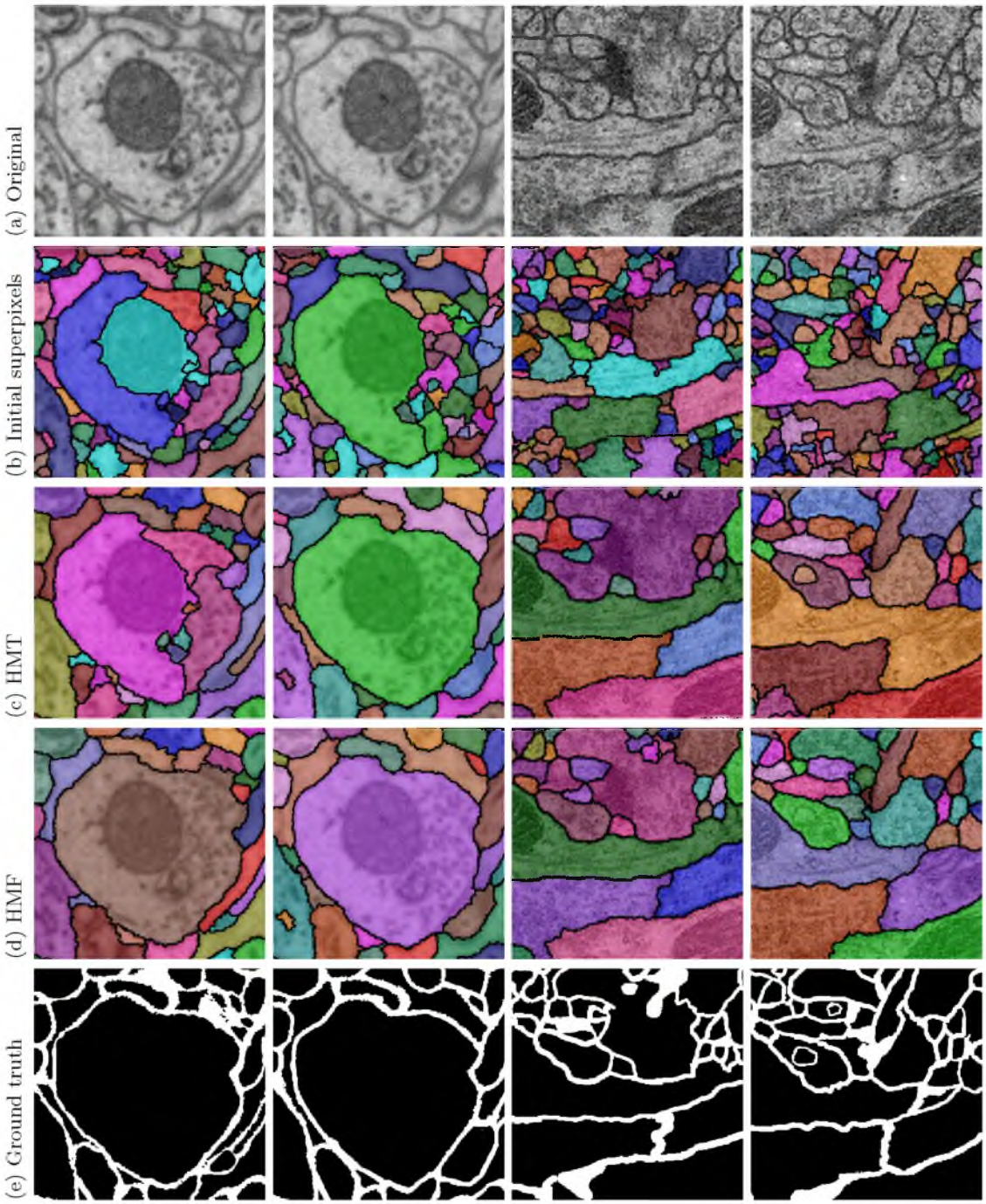


Figure 3.3: Examples of 2D segmentation results of two image regions on two consecutive sections (zoomed-in) from the mouse neuropil (the first and the second column) and the *Drosophila* VNC data set (the third and the fourth column), including (in rows) (a) original EM images, (b) initial superpixel segmentations, (c) HMT segmentation results, (d) HMF segmentation results, and (e) ground truth images. Different colors indicate individual segments.

CHAPTER 4

CONSTRAINED CONDITIONAL HIERARCHICAL MERGE TREE MODEL

In this chapter, we propose an extension to our HMT framework (Chapter 2) for object-independent natural image segmentation, which is also presented in [80]. First, we propose to formulate our merge tree as a constrained conditional model. We associate each clique that represents potential region merging with a likelihood score predicted using an ensemble boundary classifier. Final segmentations can then be efficiently inferred by finding the globally optimal labeling of the model. We call this model CCHMT. We also develop an iterative training and testing algorithm that generates various tree structures and combines them to emphasize accurate region boundaries. Experiment results and comparisons with other very recent methods on six public data sets demonstrate that our approach achieves state-of-the-art region accuracy and is very competitive in image segmentation without semantic priors.

4.1 Introduction

First, we briefly summarize edge detection works for general image segmentation. Early edge detections are mostly based on image derivatives [81, 82] or filter banks responses [83, 84]. More recent works utilize richer information such as colors and textures. One of the most notable works, gPb [5], combines multiscale local cues and globalized cues via spectral clustering and sets up a benchmark for edge detection and region segmentation research. Supervised learning techniques are a recent trend in edge detection. Ren and Bo [85] train a classifier with sparse codes on local neighborhood information and improve the edge detection performance. Dollar and Zitnick [86] propose a novel structured-learning framework using modified random decision forest for efficient edge detection. Seyedhosseini and Tasdizen [66] propose a hierarchical model to capture multiscale contextual information and achieve state-of-the-art edge detection performance.

As another category of approaches to solving image segmentation problems, early works on region segmentation seek to directly group image pixels in an unsupervised manner. Belongie et al. [6] fit Gaussian mixture models to cluster pixels based on six-dimensional color and texture features. Mean shift [8] and its variant [87] consider region segmentation as a problem of density-mode searching. A number of works belong to graph partitioning category, which regards an image as a graph with pixels being nodes and edge weights indicating dissimilarity between neighbor pixels. Normalized cuts [7] takes the image affinity matrix and partitions an image by solving eigenvalue problems. Felzenszwalb and Huttenlocher [13] greedily merge two connected components if there exists an intercomponent edge weight that is less than the largest edge weights in the minimum spanning trees of both components. Arbelaez et al. [5] propose a variant of watershed transform to generate a hierarchy of closed contours.

As in edge detection, supervised learning-based methods for region segmentation have gained popularity in recent years. This trend leads to and is further promoted by a number of publicly available computer vision data sets with human-labeled ground truth [33, 5, 88, 89, 90, 91]. Learning segmentation models from training data enables much more capability and flexibility over hand-designed/tuned unsupervised models and leads to many more interesting works.

Following the classic foreground/background segmentation, object-independent segmentation methods seek to partition an image based only on its appearance and do not utilize underlying semantics about the scene or specific information about target objects. Kim et al. propose a hypergraph-based correlation clustering framework [92] that uses structured SVM for learning the structural information from training data. Arbelaez et al. develop the multiscale combinatorial grouping (MCG) framework [93] that exploits multiscale information and uses a fast normalized cuts algorithm for region segmentation. Yu et al. [94] present a piecewise flat embedding learning algorithm and report the best published results so far on Berkeley Segmentation Data Set using the MCG framework. Two other recent superpixel-merging approaches are ISCRA [95] and GALA [60]. Starting with a fine superpixel oversegmentation, ISCRA adaptively divides the whole region merging process into different cascaded stages and trains a respective logistic regression model at each stage to determine the greedy merging, whereas GALA improves the boundary classifier training by augmenting the training set via repeatedly iterating through the merging process. Moreover, impressive results in the extensive evaluations on six public segmentation data sets are reported in [95].

Object-dependent or semantic segmentation is another branch of region segmentation. Object-dependent prior knowledge is exploited to guide or improve the segmentation process. Borenstein and Ullman [96] formulate object segmentation as a joint model that uses both low-level visual cues and high-level object class information. Some other object segmentation methods first generate object segmentation hypotheses using low-/mid-level features and then rank segments with high-level prior knowledge [97, 98]. A recent work, SCALPEL [99], incorporates high-level information in the segmentation process and can generate object proposals more efficiently and accurately. There is also a group of methods, called cosegmentation, that utilizes the homogeneity between different target objects and jointly segments multiple images simultaneously [76, 77, 78].

Our method falls into the object-independent hierarchical segmentation category. The contributions of this chapter include:

- Reformulation of the HMT model (Chapter 2) as a constrained conditional model with global optimal solutions defined and an efficient inference algorithm developed, instead of the greedy tree model.
- An iterative approach to diversify merge tree generation and improve results via segmentation accumulation.
- Experiments with state-of-the-art results on six public data sets for general image segmentation.

Compared with recent competitive hierarchical segmentation methods, ISCRA [95] and GALA [60], which use a threshold-based greedy region merging strategy, our model has two major advantages. First, the tree structure enables the incorporation of higher order image information into segmentation. The merge/split decisions are made together in a globally optimal manner instead of by looking only at local region pairs. Second, our method does not require the threshold parameter to determine when to stop merging as in ISCRA and GALA, which may be so important to the results that need carefully tuning. Furthermore, our method is almost parameter-free given the initial superpixel oversegmentation. The only parameter is the number of iterations, which can be fixed as shown in the experiments on the six data sets.

4.2 Methodology

4.2.1 Constrained conditional model

We formulate the merge tree as a constrained conditional model. It is essentially a factor graph for the merge tree, in which the node set aligns identically with $\mathcal{V}$, and each merge in the merge tree that involves three nodes $(\{v_i, v_{c_1}, v_{c_2}\}, \{e_{i,c_1}, e_{i,c_2}\})$ is considered as a clique p_i in the factor graph. We define:

- Clique p_i is at node v_i.
- Clique p_{c_1} and p_{c_2} at v_{c_1} and v_{c_2}, respectively, are the child cliques of p_i, and clique p_i is the parent clique of p_{c_1} and p_{c_2}.
- If v_i is a leaf node, $p_i = (\{v_i\}, \varnothing)$ is a leaf clique.
- Clique p_i is a nonleaf/root/nonroot clique if v_i is a nonleaf/root/nonroot node.

Figure 4.1d shows an example of the constrained conditional model factor graph of a merge tree in Figure 4.1c. The red box in Figure 4.1d shows a clique. We use the boundary label (Section 4.2.3) to indicate whether the merge at a clique happens. By assigning $y = 1$ to all leaf nodes, we denote a complete label assignment to every node in the merge tree as $\mathbf{y} = \{y_i\}_{i=1}^{|\mathcal{V}|}$. Figure 4.1d shows the set of label assignment $\mathbf{y}$ that corresponds to the final segmentation shown in Figure 4.1b.

Using the parent node index query function $\rho(\cdot)$ (Section 2.2.4), we define the *merge consistency constraint* for nonroot cliques:

$$y_i \geq y_{\rho(i)}, \forall i. \tag{4.1}$$

Clearly, for a given merge tree, a set of node labeling $\mathbf{z}$ (Section 2.2.4) subject to the region consistency constraint (2.8) can be transformed to a consistent $\mathbf{y}$ by assigning $y = 1$ to the cliques at the nodes with $z = 1$ and their descendant cliques and $y = 0$ to the rest. A consistent $\mathbf{y}$ can be transformed to $\mathbf{z}$ by assigning $z = 1$ to the nodes in $\{v_i \in \mathcal{V} \,|\, \forall i, \text{s.t. } y_i = 1 \wedge (v_i \text{ is the root} \vee y_{\rho(i)} = 0)\}$ and $z = 0$ to the rest.

We use the boundary classifier (Section 2.2.3) to predict the likelihood score $P(y \,|\, \mathbf{x})$ for each clique. An ensemble version of the boundary classifier will be introduced in Section 4.2.2. We associate each clique p_i with energy with respect to its label as

$$E_i(y_i) = -\log P(y_i),\ y_i \in \{0, 1\}. \tag{4.2}$$

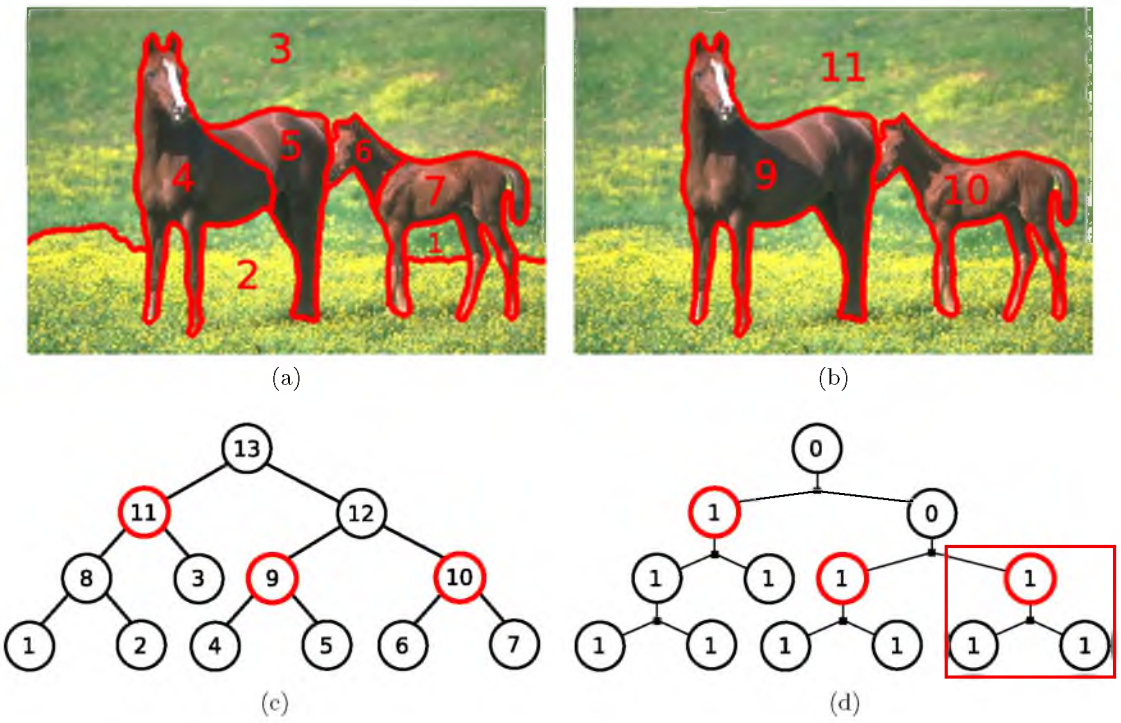


Figure 4.1: Example of (a) an initial segmentation, (b) a consistent final segmentation, (c) a merge tree, and (d) the corresponding conditional model factor graph with correct labeling. In (c), the leaf nodes have labels identical to those of the initial regions. The red nodes correspond to regions in the final segmentation. The red box in (d) indicates a clique.

Under the Markov assumption, we formulate our labeling problem as a constrained optimization problem

$$\begin{aligned} \min_{\mathbf{y}} \quad & \sum_{y_i \in \mathbf{y}} E_i(y_i),\ y_i \in \{0,1\}, \\ \text{s.t.} \quad & y_i = 1, \forall i, v_i \text{ is a leaf node}, \\ & y_i \geq y_{\rho(i)}, \forall i, v_i \text{ is a nonroot node}, \end{aligned} \tag{4.3}$$

for which we will introduce a globally optimal inference algorithm in Section 4.2.3.

4.2.2 Ensemble boundary classifier

To score each clique, we train a boundary classifier to predict the probability of each merge. To generate training labels that indicate whether the boundary between two regions exists or not, we compute the variation of information (VI) [100, 101] for both merge and split against the ground truth. The case with smaller VI deviates less from the ground truth and is adopted. See Section 2.2.3 for details about generating boundary classification labels and Section 4.3.2 for details about VI.

Boundary features and region features are extracted for classification. For a pair of merging regions, boundary features provide direct cues about how it is likely the boundary truly exists, and regional features measure geometric and textural similarities between the two regions, which can both be informative to boundary classification. We choose features mostly following [95] for comparison purposes. A summary of features is provided in Appendix A.3. The boundary classifier is not limited to any specific supervised classification model. We use random forest [67] in our experiments.

The boundary classification problem is highly nonlinear, and learning one universally accurate boundary classifier for all merging cases is essentially difficult. The size of merging regions affects the feature representativeness in classification. For instance, textural features in the form of averaged histograms among patches may not be informative when the merging regions are too small, because textural features can be extracted only from a very limited number of image patches and thus is noisy. On the other hand, when two regions are so big that they contain undersegmentation from different perceptual groups, the features again may not be meaningful, but for a different reason, that is, the histogram averaging is not able to represent the variation of textures. It is worth noting that for the same reason, different classifiers have to be learned at different merging stages in [95].

We categorize the classification problem into subproblems, train a separate subclassifier for each subproblem, and use the ensemble of the subclassifiers as the boundary classifier. We compute the median size $|s|_{\text{med}}$ of all regions observed in the training set and assign a category label to a training sample that involves regions s_i and s_j based on their sizes as in (4.4). Three subclassifiers are then trained separately using only samples with identical category labels.

$$c(s_i, s_j) = \begin{cases} 1 & \text{if } \max(|s_i|, |s_j|) < |s|_{\text{med}}, \\ 2 & \text{if } \min(|s_i|, |s_j|) < |s|_{\text{med}} \leq \max(|s_i|, |s_j|), \\ 3 & \text{otherwise.} \end{cases} \tag{4.4}$$

At testing time, a sample is categorized based on its region sizes and assigned to the corresponding subclassifier for prediction. Since all the subclassifiers are always used adjointly, we refer to the set of all subclassifiers as the boundary classifier in the rest of this chapter.

4.2.3 Inference

Exhaustive search to solve (4.3) has exponential complexity. Given the tree structure, however, we can use a bottom-up/top-down algorithm to efficiently find the exact optimal

solution under the region consistency constraint. The fundamental idea of the bottom-up/top-down algorithm is dynamic programming: in the bottom-up step, the minimum energies for both decisions (merge/split) under the constraint are kept and propagated from leaves to the root, based on which the set of best consistent decisions is made from the root to leaves in the top-down step. It is noteworthy that our bottom-up/top-down algorithm is only for inference and conceptually different from the top-down/bottom-up framework in [96], which seeks to combine high-level semantic information and low-level image features. On the other hand, the two-way message passing algorithm used in [96] and our algorithm both belong to the Pearl's belief propagation [102, 103] category, except that our inference algorithm explicitly incorporates the consistency constraint into the optimization procedure.

In the bottom-up step, a pair of energy sums are kept track of for each node v_i with children v_{c_1} and v_{c_2}: the merging energy E_i^m of node v_i and its descendants all being labeled $y = 1$ (merge), the splitting energy E_i^s of it that v_i is labeled $y_i = 0$ (split), and its descendants are labeled optimally subject to the constraint. Then, the energies can be computed bottom-up recursively as

$$E_i^m = E_{c_1}^m + E_{c_2}^m + E_i(y_i = 1), \tag{4.5}$$

$$E_i^s = \min(E_{c_1}^m, E_{c_1}^s) + \min(E_{c_2}^m, E_{c_2}^s) + E_i(y_i = 0). \tag{4.6}$$

For any leaf node v_i, we assign $E_i^m = 0$ and $E_i^s = \infty$ to enforce their being labeled $y_i = 1$. Algorithm 1 describes the bottom-up algorithm in pseudocode.

In the top-down step, we start from the root and do a depth-first search: if the merging energy of a node is lower than its splitting energy, label this node and all its descendants $y = 1$; otherwise, label this node $y = 0$ and search its children. Algorithm 2 describes the top-down algorithm in pseudocode.

Eventually, we select the set of the nodes, such that its label is $y = 1$ and its parent is labeled $y = 0$, to form an optimal final segmentation. In both algorithms, each node is visited exactly once with constant operations, and we need only linear space proportional to the number nodes for $\mathcal{T}_E$ and $\mathbf{y}$, so the time and space complexity are both $O(|\mathcal{V}|)$.

4.2.4 Iterative merge tree sampling

The performance upper bound of the hierarchical merge tree models is determined by the quality of the tree structure. If all true segments exist as nodes in the tree, they may be picked out by the inference algorithm using predictions from well-trained boundary classifiers. However, if a desirable segment is not represented by any node in the tree, the model is not able to recover the segment. Hence, the merging saliency function, which is

Algorithm 1: Bottom-up energy computation.

Input: A list of energy $\{E_i(y_i)\}_{i=1}^{|\mathcal{V}|}$ for each clique
Output: A list of energy tuples $\mathcal{T}_E = \{(E_i^m, E_i^s)\}_{i=1}^{|\mathcal{V}|}$

$\mathcal{T}_E \leftarrow \{\}$
ComputeEnergyTuples(v_r), where v_r is the root node
/ Helper function that recursively computes energy terms */*
function ComputeEnergyTuples(v_i):
    if v_i is a leaf node **then**
        $E_i^m \leftarrow 0$
        $E_i^s \leftarrow \infty$
    else
        $(E_{c_1}^m, E_{c_1}^s) \leftarrow$ **ComputeEnergyTuples**(v_{c_1})
        $(E_{c_2}^m, E_{c_2}^s) \leftarrow$ **ComputeEnergyTuples**(v_{c_2})
        $E_i^m \leftarrow E_{c_1}^m + E_{c_2}^m + E_i(y_i = 1)$
        $E_i^s \leftarrow \min(E_{c_1}^m, E_{c_1}^s) + \min(E_{c_2}^m, E_{c_2}^s) + E_i(y_i = 0)$
    end if
    $\mathcal{T}_E \leftarrow \mathcal{T}_E \cup \{(E_i^m, E_i^s)\}$
    return (E_i^m, E_i^s)
end function

Algorithm 2: Top-down label assignment.

Input: A list of energy tuples $\mathcal{T}_E = \{(E_i^m, E_i^s)\}_{i=1}^{|\mathcal{V}|}$
Output: A complete label assignment $\mathbf{y} = \{y_i\}_{i=1}^{|\mathcal{V}|}$

$\mathbf{y} \leftarrow \{\}$
AssignNodeLabels(v_r), where v_r is the root node
/ Helper function that recursively decides node labels */*
function AssignNodeLabels(v_i):
    if $E_i^m < E_i^s$ **then**
        $\mathbf{y} \leftarrow \mathbf{y} \cup \{y_i = 1\} \cup \{y_{i'} = 1 \,|\, \forall i', \text{s.t.}\, \exists k, \rho^k(i') = i\}$
    else
        $\mathbf{y} \leftarrow \mathbf{y} \cup \{y_i = 0\}$
        AssignNodeLabels(v_{c_1})
        AssignNodeLabels(v_{c_2})
    end if
end function

used to determine merging priorities, is critical to the entire performance. With a good merging saliency function, we can push the upper bound of performance and thus improve segmentation accuracy.

Statistics over the boundary strengths can be used to indicate merging saliency. We use the negated median of boundary pixel strengths as the initial representation of saliency, as mentioned in Section 2.2.2. Since a boundary classifier is essentially designed to measure the

likelihood of region merging, and it has advantages over simple boundary statistics because it takes various features from both boundary and regions, we propose to use the merging probabilities predicted by boundary classifiers as the merging saliency to construct a merge tree.

As described in Section 4.2.2, the training of a boundary classifier requires samples generated from a merge tree, but we would like to use a boundary classifier to construct a merge tree. Therefore, we propose an iterative approach that alternately collects training samples from a merge tree for the training of boundary classifiers and constructs a merge tree with the trained classifier. As illustrated in Figure 4.2a, we initially use the negated median of boundary strengths to construct a merge tree, collect region merging samples, and train a boundary classifier. Then, the boundary classifier is used to generate a new merge tree, from which new training samples are generated. We next combine the samples from the current iteration and from the previous iterations, remove duplicates, and train the next classifier. This process is repeated for a fixed number of iterations or until the segmentation accuracy on a validation set no longer improves. In practice, we fix the iteration number to 10 for all data sets. Eventually, we have a series of boundary classifiers from each training iteration. The training algorithm is described in Algorithm 3.

At testing time, we take the series of trained classifiers and iterate in a way similar to the training process, as shown in Figure 4.2b: at each iteration t, we take the previous boundary classifier f_b^{t-1} to construct a merge tree and use the current classifier f_b^t to predict each merge score in the merge tree, based on which a final segmentation S^t is inferred. Finally, we transform each segmentation into a binary closed contour map by assigning boundary pixels 1 and others 0 and average them for each image over all iterations to generate a

Algorithm 3: Iterative training algorithm.

Input: Original images $\{I_i\}_{i=1}^{N_{\text{tr}}}$, boundary maps $\{Pb_i\}_{i=1}^{N_{\text{tr}}}$, and iteration number T
Output: Boundary classifiers $\{f_b^t\}_{t=0}^{T}$

```
Generate initial superpixels {S_oi}_{i=1}^{N_tr}
for t : 0, 1, ..., T do
    if t == 0 then
        Generate {Tr_i^0}_{i=1}^{N_tr} from {S_oi}_{i=1}^{N_tr} using {Pb_i}_{i=1}^{N_tr}
    else
        Generate {Tr_i^t}_{i=1}^{N_tr} from {S_oi}_{i=1}^{N_tr} using f_b^{t-1}
    end if
    Generate samples {(X_i^t, y_i^t)}_{i=1}^{N_tr} from {Tr_i^t}_{i=1}^{N_tr}
    Train f_b^t using ∪_{t'=1}^{t} {(X_i^{t'}, y_i^{t'})}_{i=1}^{N_tr}
end for
```

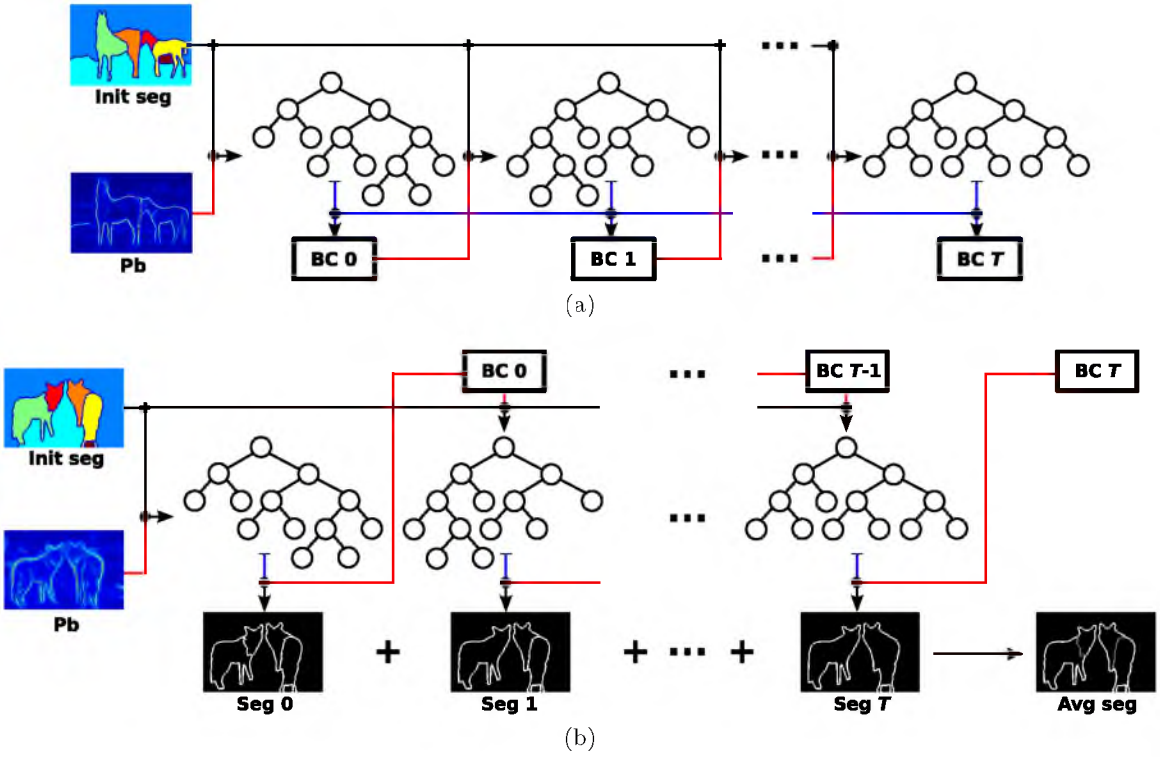


Figure 4.2: Illustrations of (a) training and (b) testing procedure of iterative merge tree sampling. Starting with fixed initial superpixels ("Init Seg"), the first iteration uses boundary probability ("Pb") statistics for merge tree generation, and the training procedure iteratively augments the training set by incorporating new samples from merge trees and trains a new boundary classifier ("BC"), which is used for merge tree generation in the next iteration. At testing time, boundary probability statistics and boundary classifiers learned at each iteration are used to generate merge trees, and each boundary classifier is used to score merge parts in the previous iteration; segmentations are generated from each merge tree and accumulated to generate the final contour hierarchy. In both figures, the black lines show the use of initial superpixels, the red lines show the use of boundary classifiers, and the blue lines show the flow of sample data collected from tree structures.

segmentation hierarchy in the form a real-valued contour map. The testing algorithm is described in Algorithm 4.

The explanation for the iterative approach is two-fold. First, by collecting samples that were not seen in previous iterations, we can explore the merge sample space and in turn explore the space of merge trees generated by the classifiers trained using the augmented sample set towards the "correct" merge tree. Second, like a bagging algorithm, segmentation averaging through iterations tends to emphasize accurate boundaries by phasing out nonsystematic errors due to incorrect tree structures or classifier mispredictions. The accumulation alleviates the difficulty of training one good classifier to generate accurate segmentations by improving via averaging.

Algorithm 4: Iterative testing algorithm.

Input: Original images $\{I_i\}_{i=1}^{N_{\text{te}}}$, boundary maps $\{Pb_i\}_{i=1}^{N_{\text{te}}}$, boundary classifiers $\{f_b^t\}_{t=0}^{T}$
Output: Hierarchical segmentation contour map $\{C_i\}_{i=1}^{N_{\text{te}}}$

```
Generate initial superpixels {S_oi}_{i=1}^{N_te}
for t : 0, 1, ..., T do
    if t == 0 then
        Generate {Tr_i^0}_{i=1}^{N_te} from {S_oi}_{i=1}^{N_te} using {Pb_i}_{i=1}^{N_te}
    else
        Generate {Tr_i^t}_{i=1}^{N_te} from {S_oi}_{i=1}^{N_te} using f_b^{t-1}
    end if
    Score merges with f_b^t and infer segmentations {S_i^t}_{i=1}^{N_te}
    Binarize {S_i^t}_{i=1}^{N_te} to contour maps {C_i^t}_{i=1}^{N_te}
end for
{C_i}_{i=1}^{N_te} = {sum_{t=0}^{T} C_i^t/(T+1)}_{i=1}^{N_te}
```

4.3 Results

We conduct experiments with two validation goals. First, we evaluate the performance of our hierarchical merge tree models with different combinations of settings. Second, we compare our method with other state-of-the-art methods.

4.3.1 Data sets

We experiment with six publicly available data sets for image segmentation:

1. Berkeley Segmentation Data Set 300 (BSDS300) [33]: 200 training and 100 testing natural images of size 481 × 321 pixels. Multiple ground truth segmentations are provided with different labeling of details.

2. Berkeley Segmentation Data Set 500 (BSDS500) [5]: an extension of BSDS300 with 200 new testing images of the same size, with multiple ground truth segmentations for each image.

3. MSRC Object Recognition Data Set (MSRC) [88]: 591 320×213 natural images with one ground truth per image. A cleaned-up version [104] is used, in which "void" regions are removed, and disconnected regions that belong to the same object class are assigned different labels in a single image.

4. PASCAL Visual Object Classes Data Set (VOC12) [90]: 1449 validation images with one ground truth per image for PASCAL VOC 2012 Challenge. The average image size is 496 × 360. We use the ground truth for object segmentation and treat the object boundary pixels as background.

5. Stanford Background Data Set (SBD) [89]: 715 320×240 images of outdoor scenes with one ground truth per image.

6. NYU Depth Data Set v2 (NYU) [91]: 1449 indoor scene images with one ground truth per image. Down-sampled versions (320×240) [85] are used with frame pixels cropped. Only RGB channels are used in our experiment; the depth maps are not used.

In order to compare with the other state-of-the-art methods, we follow [95] and train our boundary classifiers with the 200 training images in BSDS300. Five ground truth segmentations are selected for each image in the order of increasing details as indicated by the number of true segments. The training and the testing are done for each detail level, and the results are combined into a segmentation hierarchy. In our performance evaluation of different configurations of the merge tree model, we test on the testing images in BSDS500. For comparisons with other methods, we test on all six data sets.

4.3.2 Evaluation metrics

Following [5], we use the segmentation covering [90], the probabilistic Rand index [105], and the variation of information [100, 101] for segmentation accuracy evaluation. Here, we summarize the three evaluation metrics. For more details, please refer to [5].

The segmentation covering measures averaged matching between proposed segments with a ground truth labeling, defined as

$$SC(S, S_g) = \sum_{s_i \in S} \frac{|s_i|}{|\mathcal{P}|} \max_{s_j \in S_g} \frac{|s_i \cap s_j|}{|s_i \cup s_j|}, \tag{4.7}$$

where $\mathcal{P}$ is the set of all pixels in an image. It matches each proposed segment to a true segment, with which the proposed segment has the largest overlapping ratio, and computes the sum of such optimal overlapping ratios weighted by relative segment sizes.

The Rand index, originally proposed in [71], measures pairwise similarity between two multilabel clusterings. It is defined as the ratio of the number of pixel pairs that have identical labels in S and S_g or have different labels in S and S_g, over the number of all pixel pairs.

$$RI(S, S_g) = \frac{1}{\binom{|\mathcal{P}|}{2}} \sum_{i<j} \mathbb{I}\left(S(i) = S(j) \wedge S_g(i) = S_g(j)\right), \tag{4.8}$$

where $S(i)$ is the label of the i-th pixel in S, and $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the input condition is met or 0 otherwise. The probabilistic Rand index is the Rand index averaged over multiple ground truth labelings if available.

The variation of information measures the relative entropy between a proposed segmentation and a ground truth labeling, defined as

$$VI(S, S_g) = H(S \mid S_g) + H(S_g \mid S), \tag{4.9}$$

where $H(S \mid S_g)$ and $H(S_g \mid S)$ are conditional image entropies. Denoting the set of all labels in S as $\mathcal{L}_S$ and the set of all labels in S_g as $\mathcal{L}_{S_g}$, we have

$$H(S \mid S_g) = \sum_{\substack{l \in \mathcal{L}_S, \\ l_g \in \mathcal{L}_{S_g}}} P(l, l_g) \log \frac{P(l_g)}{P(l, l_g)}, \tag{4.10}$$

where $P(l_g)$ is the probability that a pixel in S_g receives label l_g, and $P(l, l_g)$ is the joint probability that a pixel receives label l in S and label l_g in S_g. $H(S_g \mid S)$ can be defined similarly by switching S and S_g in (4.10).

For each data set, segmentation results are evaluated at a universal fixed scale (ODS) for the entire data set and at a fixed scale per testing image (OIS), following [5]. The evaluated numbers are averaged over all available ground truth labelings. As pointed out in [95], since we focus on region segmentation, the pixelwise boundary-based evaluations for contour detection results [5] are not relevant, and we use only the region-based metrics.

4.3.3 Experiments

We use the watershed algorithm for superpixel generation, for which the water level needs to be specified. In general, lowering the water level reduces undersegmentation by producing more superpixels, which gives us sets of high-precision superpixels to start with, but also increases the computation cost. We fixed the water level at $\theta_w = 0.01$ for all five data sets (BSDS300/500, MSRC, SBD, and VOC12), except the NYU data set. For the NYU data set of indoor scene images, we observe the decrease in gPb boundary detection strength, so we lower the water level to $\theta_w = 0.001$. We also premerge regions smaller than $\theta_p^{a_1} = 20$ pixels to their neighboring regions with the lowest boundary barrier, i.e.,the median of boundary detection probabilities on the boundary pixels between the two regions. We train $\theta_{\text{rf}}^t = 255$ fully grown decision trees for the random forest boundary classifier. To train each decision tree, $\theta_{\text{rf}}^s = 70\%$ of training samples are randomly drawn and used. The number of features examined at each node is the square root of the total number of features. In the experiments, the training data are usually imbalanced. The ratios between the number of positive and negative samples are sometimes considerably greater than 1. Therefore, we assign to each class a weight reciprocal to the number of samples in the class

to balance the training. We fix the number of iterations to 10 for all data sets for the iterative merge tree sampling.

Appendix A.3 summarizes the features used for boundary classification.

4.3.3.1 Ensemble vs. single boundary classifier and constrained conditional model vs. greedy tree model

We evaluate the performance of using single ("SC") or ensemble boundary classifiers ("EC") (Section 4.2.2) compared with our hierarchical merge tree models. We also compare the proposed constrained conditional model ("CCM") formulation and greedy tree model ("Greedy") in HMT (Chapter 2). The training is done using the 200 training images in BSDS300 as described in Section 4.3.1, and we show the testing results on the 200 testing images in BSDS500 in Table 4.1.

A comparison between the first two rows in Table 4.1 shows that using ensemble boundary classifiers outperforms using only a single boundary classifier among all metrics, which supports our claim that the classifier ensemble is better able to capture underlying merging characteristics of regions at different size scales.

Comparing the first and the third row, we can see that CCM significantly outperforms the greedy model in terms of VI, which is preferred over the other metrics for segmentation quality evaluation [60]. It appears that CCM is outperformed by the greedy tree model in terms of PRI, but this is because both models are trained using the labels determined based on VI (Section 4.2.2). We perform another experiment where both are trained using the labels determined based on the Rand index, and CCM outperforms the greedy model 0.829 vs. 0.826 in terms of ODS PRI and 0.855 vs. 0.848 in terms of OIS PRI.

The fourth row shows the results using the HMT model. It is clear that the proposed

Table 4.1: Segmentation results of BSDS500 using the constrained conditional model (CCM) formulation or greedy tree model (Greedy) in combination with the ensemble boundary classifier (EC) or single boundary classifier (SC). The segmentation covering (Covering), the probabilistic Rand index (PRI), and the variation of information (VI) are reported for optimal data set scale (ODS) and optimal image scale (OIS).

	Covering		PRI		VI	
HMT variant	ODS	OIS	ODS	OIS	ODS	OIS
CCM+EC	0.594	0.607	0.804	0.809	1.682	1.556
CCM+SC	0.573	0.581	0.779	0.781	1.690	1.617
Greedy+EC	0.587	0.620	0.821	0.834	1.737	1.589
Greedy+SC [41]	0.582	0.601	0.805	0.812	1.748	1.639

constrained conditional model and ensemble boundary classifier (CCHMT) are an improvement over our previous approach without including the iterative segmentation merge tree sampling.

4.3.3.2 CCHMT without vs. with iterative merge tree sampling

We evaluate the performance of the CCHMT model with or without iterative merge tree sampling (Section 4.2.4). The experimental setting follows the previous experiments in Section 4.3.3.1. The constrained conditional model formulation and ensemble boundary classifiers are adopted. The testing results at each iteration are shown in Table 4.2.

We can see that despite occasional oscillations, the results are improved through iterations. The rate of improvement slows down as more iterations are included in the averaging process. More sophisticated ways of choosing segmentations to average over can be used, such as to average segmentations only from the iterations that achieve the top accuracy on some validation set. In our experiment, since we would like to compare our method with other methods, we keep the same setting for training and testing data sets and do not use a separate validation set. We fix the iteration number to 10 and only report the results from averaging all the segmentations.

Table 4.2: Segmentation results of BSDS500 using CCHMT without (Iteration 0) and with (Iteration 1 to 10) iterative merge tree sampling and segmentation accumulation. The segmentation covering (Covering), the probabilistic Rand index (PRI), and the variation of information (VI) are reported for optimal data set scale (ODS) and optimal image scale (OIS).

	Covering		PRI		VI	
Iteration	ODS	OIS	ODS	OIS	ODS	OIS
0	**0.594**	**0.607**	**0.804**	**0.809**	**1.682**	**1.556**
1	0.601	0.637	0.825	0.841	1.661	1.498
2	0.612	0.654	0.829	0.853	1.596	1.432
3	0.618	0.666	0.834	0.860	1.564	1.407
4	0.624	0.671	0.834	0.864	1.545	1.391
5	0.624	0.676	0.836	0.865	1.544	1.378
6	0.626	0.678	0.835	0.867	1.539	1.374
7	0.628	0.679	0.835	0.868	1.532	1.373
8	0.628	0.679	0.835	0.869	1.534	1.370
9	0.628	0.680	0.835	0.869	1.530	1.371
10	**0.629**	**0.679**	**0.835**	**0.869**	**1.526**	**1.375**

4.3.3.3 Comparisons with other methods

In this section, we compare our proposed iterative CCHMT (CCM + ensemble boundary classifier + iteration) with various other state-of-the-art region segmentation methods and benchmarks [5, 95, 60, 93, 106, 92, 94] in very recent years on the public data sets. The testing results are shown in Table 4.3. Note that [92] generates a single segmentation instead of contour hierarchies for each image. The OIS evaluations are therefore essentially the same as the ODS results, so we exclude the OIS entries for the sake of clarity.

From Table 4.3, we can see that our method is highly competitive and outperforms very recent state-of-the-art methods on some data sets, including BSDS500, which is the most used data set for image segmentation evaluation. It is noteworthy that the generalization of our method is almost as good as ISCRA [95] by being trained only on BSDS (general natural photos) and achieving competitive results on the NYU data set (indoor scene photos). It is also worth pointing out that our hierarchical segmentation framework can be used in combination with other features that can better guide the boundary classification. For example, using the most recent piecewise flat embedding (PFE) [94], we expect the results to be further improved in a manner similarly to the results from "MCG" to "PFE-MCG" on BSDS500 in Table 4.3. Figure 4.3 shows sample testing segmentation results for each data set.

4.4 Conclusion

In this chapter, we proposed a constrained conditional formulation to the HMT model for natural image segmentation. Globally optimal solutions can be efficiently found under constraints to generate final segmentations thanks to the tree structure. We also introduced a modification to the model that iteratively trains a new boundary classifier with accumulated samples for merge tree construction and merging probability prediction and accumulates segmentation to generate hierarchical contour maps. For further improvement, the combination of merge trees from each iteration as one single model and its global resolution can be investigated. Furthermore, it would be interesting to study the application of our method to semantic segmentation with the introduction of object-dependent prior knowledge.

Table 4.3: Segmentation results of (a) BSDS300, (b) BSDS500, (c) MSRC, (d) VOC12, (e) SBD, and (f) NYU data set, using different methods. The segmentation covering (Covering), the probabilistic Rand index (PRI), and the variation of information (VI) are reported for optimal data set scale (ODS) and optimal image scale (OIS).

(a) BSDS300

	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.59	0.65	0.81	0.85	1.65	1.47
ISCRA [95]	0.60	**0.67**	0.81	**0.86**	1.61	1.40
HOCC [92]	0.60	-	0.81	-	1.74	-
MCG [93]	**0.61**	**0.67**	0.81	**0.86**	**1.55**	**1.37**
Ours	**0.61**	**0.67**	**0.82**	**0.86**	1.58	1.40

(b) BSDS500

	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.59	0.65	0.83	0.86	1.69	1.48
ISCRA [95]	0.59	0.66	0.82	0.86	1.60	1.42
GALA [60]	0.61	0.67	**0.84**	0.86	1.56	**1.36**
HOCC [92]	0.60	-	0.83	-	1.79	-
DC [106]	0.59	0.64	0.82	0.85	1.68	1.54
MCG [93]	0.61	0.66	0.83	0.86	1.57	1.39
PFE-mPb [94]	0.62	0.67	**0.84**	0.86	1.61	1.43
PFE-MCG [94]	0.62	**0.68**	**0.84**	**0.87**	1.56	**1.36**
Ours	**0.63**	**0.68**	**0.84**	**0.87**	**1.53**	1.38

(c) MSRC

	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.65	0.75	0.78	0.85	1.28	0.99
ISCRA [95]	**0.67**	0.75	0.77	0.85	**1.18**	1.02
Ours	**0.67**	**0.77**	**0.79**	**0.86**	1.23	**0.93**

Table 4.3: (Continued).

(d) VOC12

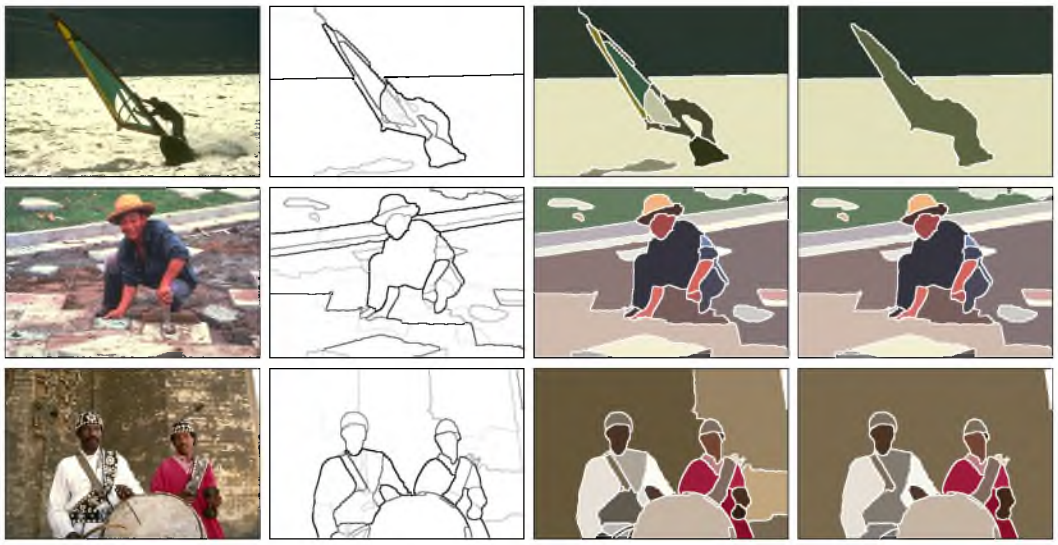
	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.46	0.59	0.76	0.88	0.65	0.50
ISCRA [95]	**0.50**	0.58	0.69	0.75	1.01	0.93
Ours	0.49	**0.63**	**0.77**	**0.91**	**0.60**	**0.44**

(e) SBD

	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.58	0.64	0.86	0.89	1.88	1.62
ISCRA [95]	**0.62**	**0.68**	**0.87**	**0.90**	1.73	1.49
Ours	0.61	0.67	0.86	**0.90**	**1.72**	**1.48**

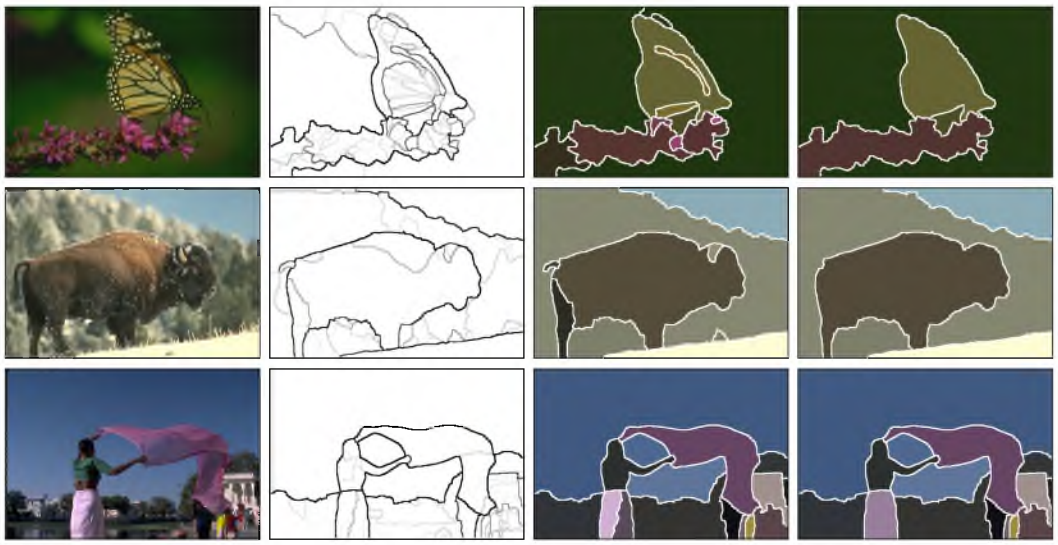
(f) NYU

	Covering		PRI		VI	
Method	ODS	OIS	ODS	OIS	ODS	OIS
gPb-OWT-UCM [5]	0.55	0.60	**0.90**	**0.92**	1.89	1.89
ISCRA [95]	**0.57**	**0.62**	**0.90**	**0.92**	**1.82**	**1.63**
Ours	**0.57**	0.61	**0.90**	**0.92**	1.83	1.66



(i) Original (ii) Contours (iii) ODS (iv) OIS

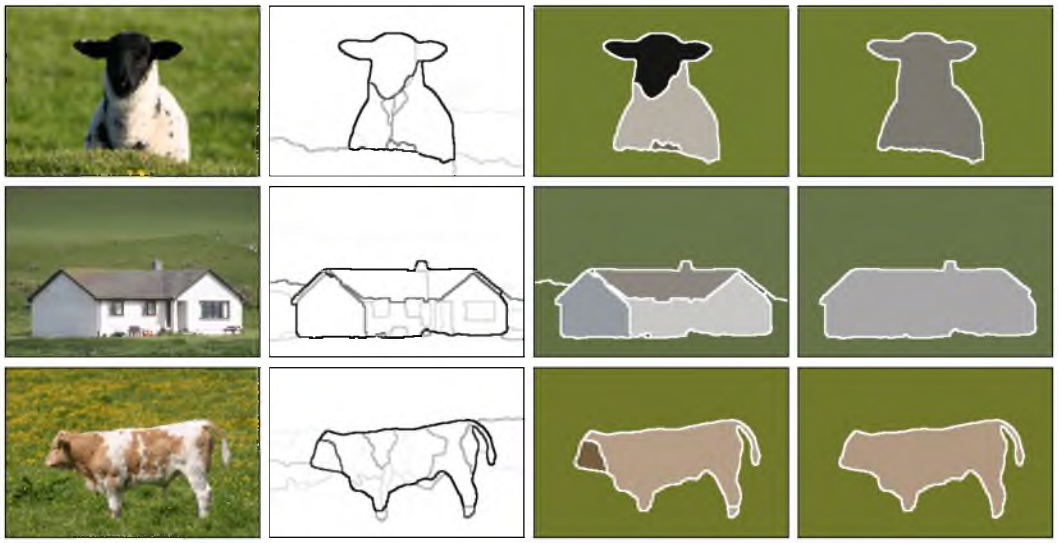
(a) BSDS300



(i) Original (ii) Contours (iii) ODS (iv) OIS

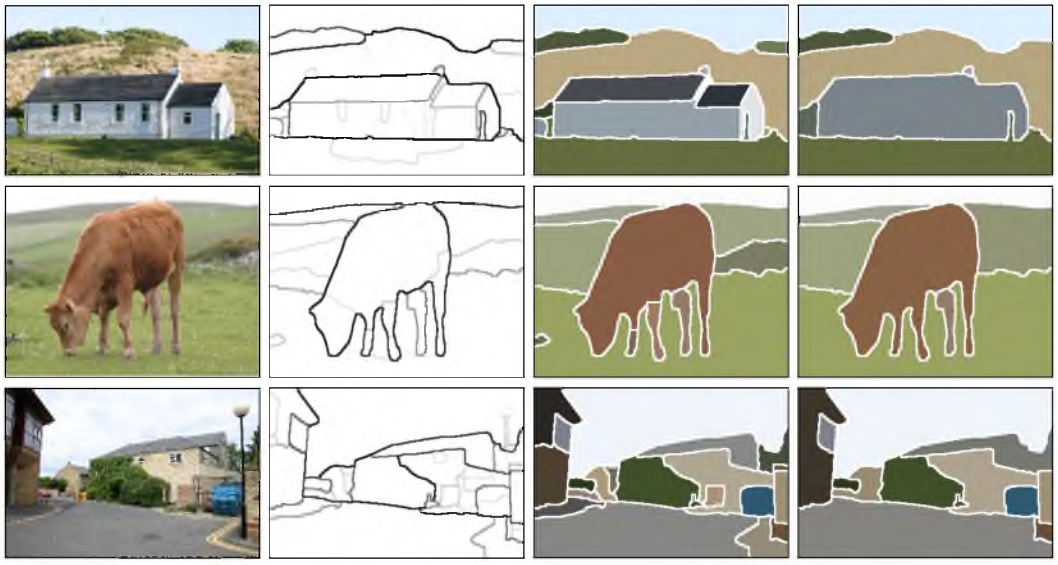
(b) BSDS500

Figure 4.3: Examples of segmentation results of (a) BSDS300, (b) BSDS500, (c) MSRC, (d) VOC12, (e) SBD, and (f) NYU data set, including (in columns) (i) original images, (ii) hierarchical contour maps, (iii) ODS covering segmentations, and (iv) OIS covering segmentations. The training uses BSDS300 training images. Contours are thickened for visualization purposes.



(i) Original (ii) Contours (iii) ODS (iv) OIS

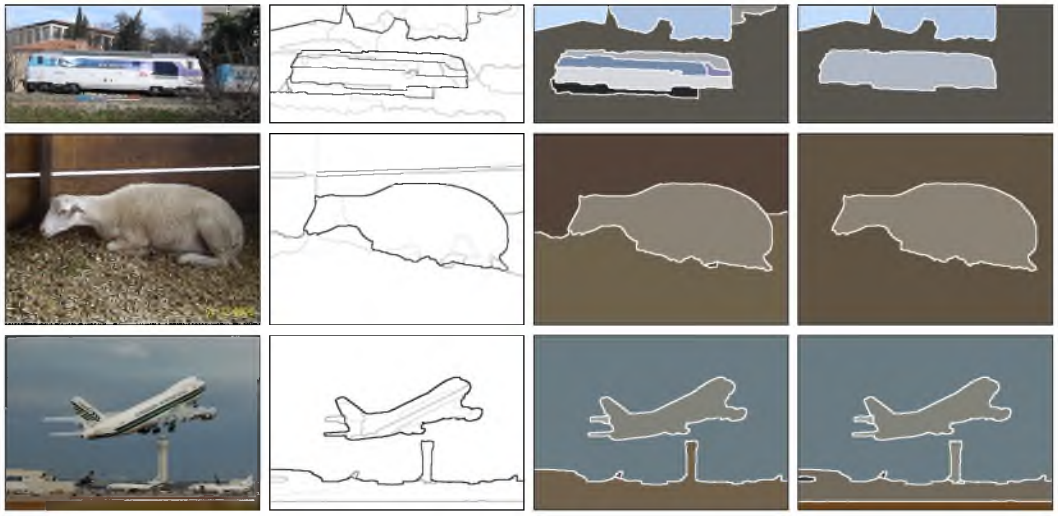
(c) MSRC



(i) Original (ii) Contours (iii) ODS (iv) OIS

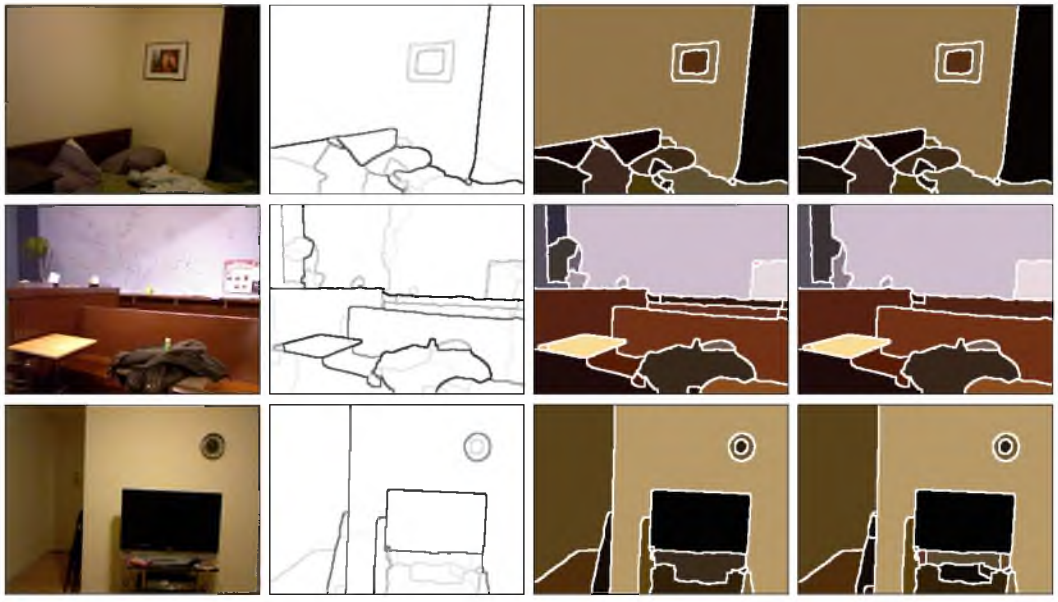
(d) VOC12

Figure 4.3: (Continued).



(i) Original (ii) Contours (iii) ODS (iv) OIS

(e) SBD



(i) Original (ii) Contours (iii) ODS (iv) OIS

(f) NYU

Figure 4.3: (Continued).

CHAPTER 5

SEMI-SUPERVISED HIERARCHICAL MERGE TREE MODEL

Given initial superpixels and structures, the performance of HMT (Chapter 2) as well as most other superpixel-merging image segmentation methods largely depends on accurate boundary predictions, which are determined by a boundary classification function. Such functions are usually learned with supervised algorithms that demand considerable ground truth data. In this chapter, we propose a semi-supervised learning-based HMT model, named SSHMT, to reduce this demand. We focus on the application of SSHMT to EM image segmentation, for which the collecting of ground truth data is extremely laborious and requires expertise. Based on the merge tree structure, we develop a differentiable unsupervised loss term that enforces consistent boundary predictions. We then propose a Bayesian model that combines the supervised and the unsupervised information for probabilistic learning of the boundary classification function. The experimental results on three EM data sets demonstrate that by using a subset of only 3% to 7% of the entire ground truth data, SSHMT consistently performs close to fully supervised HMT with full labeled data sets, and significantly outperforms fully supervised HMT with the same labeled subset.

5.1 Introduction

Similar to the boundary detection/region segmentation pipeline for natural image segmentation [5, 95, 93, 80], most recent EM image segmentation methods use a membrane detection/cell segmentation pipeline. First, a membrane detector generates pixelwise confidence maps of membrane predictions using local image cues [107, 46, 66]. Next, region-based methods are applied to transform the membrane confidence maps into cell segments. It has been shown that region-based methods are necessary for improving the segmentation accuracy from membrane detections for EM images [39]. A common approach to region-based segmentation is to transform a membrane confidence map into oversegmenting

superpixels and use them as "building blocks" for final segmentation. To correctly combine superpixels, greedy region agglomeration based on certain boundary saliency has been shown to work [60]. Meanwhile, structures, such as loopy graphs [108, 109] or trees [41, 110, 111], are more often imposed to represent the region merging hierarchy and help transform the superpixel combination search into graph labeling problems. To this end, local [41, 109] or structured [110, 111] learning-based methods are developed.

Most current region-based segmentation methods use a scoring function to determine how likely it is that two adjacent regions should be combined. Such scoring functions are usually learned in a supervised manner that demands a considerable amount of high-quality ground truth data. Obtaining such ground truth data, however, involves manual labeling of image pixels and is very labor intensive, especially given the large scale and complex structures of EM images. To alleviate this demand, Parag et al. have recently proposed an active learning framework [112, 113] that starts with small sets of labeled samples and constantly measures the disagreement between a supervised classifier and a semi-supervised label propagation algorithm on unlabeled samples. Only the most disagreed samples are pushed to users for interactive labeling. The authors demonstrate that by using 15% to 20% of all labeled samples, the method can perform similar to the underlying fully supervised method with a full training set. One disadvantage of this framework is that it does not directly explore the unsupervised information while searching for the optimal classification function. Also, retraining is required for the supervised algorithm at each iteration, which can be time consuming especially when more iterations with fewer samples per iteration are used to maximize the utilization of supervised information and minimize human effort. Moreover, repeated human interactions may lead to extra cost overhead in practice.

In this chapter, we propose a semi-supervised learning framework for region-based neuron segmentation that seeks to reduce the demand for labeled data by exploiting the underlying correlation between unsupervised data samples. Based on the merge tree structure [41, 110, 111], we redefine the labeling constraint and formulate it into a differentiable loss function that can be effectively used to guide the unsupervised search in the function hypothesis space. We then develop a Bayesian model that incorporates both unsupervised and supervised information for probabilistic learning. The parameters that are essential to balancing the learning can be estimated from the data automatically. Our method works with a very small amount of supervised data and requires no further human interaction. We show that by using only 3% to 7% of the labeled data, our method performs consistently close to the state-of-the-art fully supervised algorithm with entire supervised data sets

(Section 5.3). Also, our method can be conveniently adopted to replace the supervised algorithm in the active learning framework [112, 113] and further improve the overall segmentation performance.

5.2 Methodology

5.2.1 Merge consistency constraint

Following the HMT notations, we define a clique path of length L that starts at clique p_i as an ordered set $\boldsymbol{\pi}_i^L = \{p_{\rho^l(i)}\}_{l=0}^{L-1}$. Based on the merge consistency constraint (4.1), we have

Theorem 1 *Any consistent label sequence* $\mathbf{y}_i^L = \{y_{\rho^l(i)}\}_{l=0}^{L-1}$ *for* $\boldsymbol{\pi}_i^L$ *under the merge consistency constraint is monotonically nonincreasing.*

Proof: Assume there exists a label sequence $\mathbf{y}_i^L$ subject to the merge consistency constraint that is not monotonically nonincreasing. By definition, there must exist $k \geq 0$, s.t. $y_{\rho^k(i)} < y_{\rho^{k+1}(i)}$. Let $j = \rho^k(i)$, then $\rho^{k+1}(i) = \rho(j)$, and thus $y_j < y_{\rho(j)}$. This violates the merge consistency constraint (4.1), which contradicts the initial assumption that $\mathbf{y}_i^L$ is subject to the merge consistency constraint. Therefore, the initial assumption must be false, and all label sequences that are subject to the merge consistency constraint must be monotonically nonincreasing.

□

Intuitively, Theorem 1 states that while moving up in a merge tree, once a split occurs, no merge shall occur again among the ancestor cliques in that path. As an example, a consistent label sequence for the clique path $\{p_8, p_{11}, p_{13}\}$ in Figure 4.1c can only be $\{y_8, y_{11}, y_{13}\} = \{0,0,0\}$, $\{1,0,0\}$, $\{1,1,0\}$, or $\{1,1,1\}$. Any other label sequence, such as $\{1,0,1\}$, is not consistent. In contrast to the region consistency constraint (2.8), the merge consistency constraint is a local constraint that holds for the entire leaf-to-root clique paths as well as any of their subparts, which allows certain computations to be decomposed as shown later in Section 5.3.2.

Let f_i be a predicate that denotes whether $y_i = 1$ for clique p_i. We can express the nonincreasing monotonicity of any consistent label sequence for $\boldsymbol{\pi}_i^L$ in disjunctive normal form (DNF) as

$$F_i^L = \bigvee_{j=0}^{L} \left(\bigwedge_{k=0}^{j-1} f_{\rho^k(i)} \wedge \bigwedge_{k=j}^{L-1} \neg f_{\rho^k(i)} \right), \tag{5.1}$$

which always holds *true* according to Theorem 1. We approximate F_i^L with real-valued variables and operators by replacing *true* with 1, $false$ with 0, and f with real-valued $\tilde{f}$. A

negation $\neg f$ is replaced by $1 - \tilde{f}$; conjunctions are replaced by multiplications; disjunctions are transformed into negations of conjunctions using De Morgan's laws and then replaced. The real-valued DNF approximation is

$$\tilde{F}_i^L = 1 - \prod_{j=0}^{L} \left(1 - \prod_{k=0}^{j-1} \tilde{f}_{\rho^k(i)} \cdot \prod_{k=j}^{L-1} \left(1 - \tilde{f}_{\rho^k(i)} \right) \right), \tag{5.2}$$

which is valued 1 for any consistent label assignments. Observing $\tilde{f}$ is exactly a binary boundary classifier (Section 2.2.3), we further relax it to be a classification function that predicts $P(y = 1 \,|\, \mathbf{x}) \in [0, 1]$. The choice of $\tilde{f}$ can be arbitrary as long as it is (piecewise) differentiable (Section 5.2.2). In this work, we use a logistic sigmoid function with a linear discriminant

$$\tilde{f}(\mathbf{x}; \boldsymbol{w}) = \frac{1}{1 + \exp(-\boldsymbol{w}^\top \mathbf{x})}, \tag{5.3}$$

which is parameterized by $\boldsymbol{w}$.

We would like to find an $\tilde{f}$ so that its predictions satisfy the DNF (5.2) for any path in a merge tree. We will introduce the learning of such $\tilde{f}$ in a semi-supervised manner in Section 5.2.2.

5.2.2 Bayesian semi-supervised learning model

To learn the boundary classification function $\tilde{f}$, we use both supervised and unsupervised data. Supervised data are the clique samples with labels that are generated from ground truth segmentations. Unsupervised samples are those for which we do not have labels. They can be from the images for which we do not have the ground truth or wish to segment. We use $\mathbf{X}_s$ to denote the collection of supervised sample feature vectors and $\mathbf{y}_s$ for their true labels. $\mathbf{X}$ is the collection of all supervised and unsupervised samples.

Let $\tilde{\boldsymbol{f}}_{\boldsymbol{w}} = [\tilde{f}_{j_1}, \ldots, \tilde{f}_{j_{N_s}}]^\top$ be the predictions about the supervised samples in $\mathbf{X}_s$, and $\tilde{\boldsymbol{F}}_{\boldsymbol{w}} = [\tilde{F}_{i_1}^L, \ldots, \tilde{F}_{i_{N_u}}^L]^\top$ be the DNF values (5.2) for all paths from $\mathbf{X}$. We are now ready to build a probabilistic model that includes a regularization prior, an unsupervised likelihood, and a supervised likelihood.

The prior is an i.i.d. Gaussian $\mathcal{N}(0, 1)$ that regularizes $\boldsymbol{w}$ to prevent overfitting. The unsupervised likelihood is an i.i.d. Gaussian $\mathcal{N}(0, \sigma_u)$ on the differences between each element of $\tilde{\boldsymbol{F}}_{\boldsymbol{w}}$ and 1. It requires the predictions of $\tilde{f}$ to conform the merge consistency constraint for every path. Maximizing the unsupervised likelihood allows us to narrow down the potential solutions to a subset in the classifier hypothesis space without label information by exploring the sample feature representation commonality. The supervised likelihood is an i.i.d. Gaussian $\mathcal{N}(0, \sigma_s)$ on the prediction errors for supervised samples to

enforce accurate predictions. It helps avoid consistent but trivial solutions of $\tilde{f}$, such as the ones that always predict $y = 1$ or $y = 0$, and guides the search towards the correct solution. The standard deviation parameters σ_u and σ_s control the contributions of the three terms. They can be preset to reflect our prior knowledge about the model distributions, tuned using a holdout set, or estimated from data.

By applying Bayes' rule, we have the posterior distribution of $\boldsymbol{w}$ as

$$\begin{aligned} P(\boldsymbol{w} \mid \mathbf{X}, \mathbf{X}_s, \mathbf{y}_s, \sigma_u, \sigma_s) &\propto P(\boldsymbol{w}) \cdot P(\mathbf{1} \mid \mathbf{X}, \boldsymbol{w}, \sigma_u) \cdot P(\mathbf{y}_s \mid \mathbf{X}_s, \boldsymbol{w}, \sigma_s) \\ &\propto \exp\left(-\frac{\|\boldsymbol{w}\|_2^2}{2}\right) \\ &\quad \cdot \frac{1}{\left(\sqrt{2\pi}\sigma_u\right)^{N_u}} \exp\left(-\frac{\|\mathbf{1} - \tilde{\boldsymbol{F}}_{\boldsymbol{w}}\|_2^2}{2\sigma_u^2}\right) \\ &\quad \cdot \frac{1}{\left(\sqrt{2\pi}\sigma_s\right)^{N_s}} \exp\left(-\frac{\|\mathbf{y}_s - \tilde{\boldsymbol{f}}_{\boldsymbol{w}}\|_2^2}{2\sigma_s^2}\right), \end{aligned} \tag{5.4}$$

where N_u and N_s are the number of elements in $\tilde{\boldsymbol{F}}_{\boldsymbol{w}}$ and $\tilde{\boldsymbol{f}}_{\boldsymbol{w}}$, respectively; $\mathbf{1}$ is a N_u-dimensional vector of ones.

5.2.3 Inference

We infer the model parameters $\boldsymbol{w}$, σ_u, and σ_s using maximum a posteriori estimation. We effectively minimize the negative logarithm of the posterior

$$\begin{aligned} J(\boldsymbol{w}, \sigma_u, \sigma_s) =& \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{2\sigma_u^2}\|\mathbf{1} - \tilde{\boldsymbol{F}}_{\boldsymbol{w}}\|_2^2 + N_u \log \sigma_u \\ &+ \frac{1}{2\sigma_s^2}\|\mathbf{y}_s - \tilde{\boldsymbol{f}}_{\boldsymbol{w}}\|_2^2 + N_s \log \sigma_s. \end{aligned} \tag{5.5}$$

Observe that the DNF formula in (5.2) is differentiable. With any (piecewise) differentiable choice of $\tilde{f}_{\boldsymbol{w}}$, we can minimize (5.5) using (sub-) gradient descent. The gradient of (5.5) with respect to the classifier parameter $\boldsymbol{w}$ is

$$\nabla_{\boldsymbol{w}} J = \boldsymbol{w}^\top - \frac{1}{\sigma_u^2}\left(\mathbf{1} - \tilde{\boldsymbol{F}}_{\boldsymbol{w}}\right)^\top \nabla_{\boldsymbol{w}} \tilde{\boldsymbol{F}}_{\boldsymbol{w}} - \frac{1}{\sigma_s^2}\left(\mathbf{y}_s - \tilde{\boldsymbol{f}}_{\boldsymbol{w}}\right)^\top \nabla_{\boldsymbol{w}} \tilde{\boldsymbol{f}}_{\boldsymbol{w}}, \tag{5.6}$$

Since we choose $\tilde{f}$ to be a logistic sigmoid function with a linear discriminant (5.3), the j-th ($j = 1, \ldots, N_s$) row of $\nabla_{\boldsymbol{w}} \tilde{\boldsymbol{f}}_{\boldsymbol{w}}$ is

$$\nabla_{\boldsymbol{w}} \tilde{f}_j = \tilde{f}_j(1 - \tilde{f}_j) \cdot \mathbf{x}_j^\top. \tag{5.7}$$

where $\mathbf{x}_j$ is the j-th element in $\mathbf{X}_s$.

Defining $g_j = \prod_{k=0}^{j-1} \tilde{f}_{\rho^k(i)} \cdot \prod_{k=j}^{L-1}(1 - \tilde{f}_{\rho^k(i)})$, $j = 0, \ldots, L$, we write (5.2) as $\tilde{F}_i^L = 1 - \prod_{j=0}^{L}(1 - g_j)$ as the i-th ($i = 1, \ldots, N_u$) element of $\tilde{\boldsymbol{F}}_{\boldsymbol{w}}$. Then, the i-th row of $\nabla_{\boldsymbol{w}} \tilde{\boldsymbol{F}}_{\boldsymbol{w}}$ is

$$\nabla_{\boldsymbol{w}} \tilde{F}_i^L = \sum_{j=0}^{L} \left(g_j \prod_{\substack{k=0 \\ k \neq j}}^{L} (1 - g_k) \right) \left(\sum_{k=0}^{j-1} \frac{\nabla_{\boldsymbol{w}} \tilde{f}_{\rho^k(i)}}{\tilde{f}_{\rho^k(i)}} - \sum_{k=j}^{L-1} \frac{\nabla_{\boldsymbol{w}} \tilde{f}_{\rho^k(i)}}{1 - \tilde{f}_{\rho^k(i)}} \right), \tag{5.8}$$

where $\nabla_{\boldsymbol{w}} \tilde{f}_{\rho^k(i)}$ can be computed using (5.7).

We also alternately estimate σ_u and σ_s along with $\boldsymbol{w}$. Setting $\nabla_{\sigma_u} J = 0$ and $\nabla_{\sigma_s} J = 0$, we update σ_u and σ_s using the closed-form solutions

$$\sigma_u = \frac{\|\mathbf{1} - \tilde{\boldsymbol{F}}_{\boldsymbol{w}}\|_2}{\sqrt{N_u}} \tag{5.9}$$

$$\sigma_s = \frac{\|\mathbf{y}_s - \tilde{\boldsymbol{f}}_{\boldsymbol{w}}\|_2}{\sqrt{N_s}}. \tag{5.10}$$

At testing time, we apply the learned $\tilde{f}$ to testing samples to predict their merging likelihood. Eventually, we compute the node potentials with (2.9) and apply the greedy inference algorithm to acquire the final node label assignment (Section 2.2.4).

5.3 Results

We validate the proposed algorithm for 2D and 3D segmentation of neurons in three EM image data sets. For each data set, we apply SSHMT to the same segmentation tasks using different amounts of randomly selected subsets of ground truth data as the supervised sets.

5.3.1 Data sets

5.3.1.1 Mouse neuropil data set

The mouse neuropil data set we use here is the same as that in Section 2.3.1.2. We also follow the same splitting setting in Section 2.3.1.2 that uses 14 images with ground truth as the whole supervised data set and the other 56 images for testing. We test our algorithm using 14 (100%), 7 (50%), 3 (21.42%), 2 (14.29%), 1 (7.143%), and half (3.571%) ground truth image(s) as the supervised data. We use all 70 images as the unsupervised data for training. We target 2D segmentation for this data set.

5.3.1.2 Mouse cortex data set

The mouse cortex data set we use here is the $1024 \times 1024 \times 100$ training stack of that in Section 2.3.1.3. We use the first $1024 \times 1024 \times 50$ substack as the supervised set and the second $1024 \times 1024 \times 50$ substack for testing. There are 327 ground truth neuron segments

that are larger than 1000 pixels in the supervised substack, which we consider as all the available supervised data. We test the performance of our algorithm by using 327 (100%), 163 (49.85%), 81 (24.77%), 40 (12.23%), 20 (6.116%), 10 (3.058%), and 5 (1.529%) true segments. Both the supervised and the testing substack are used for the unsupervised term. Due to the unavailability of the ground truth data, we did not experiment with the original testing image stack from the challenge. We target 3D segmentation for this data set.

5.3.1.3 *Drosophila* neuropil data set

The *Drosophila melanogaster* larval neuropil [27] is a $500 \times 500 \times 500$ FIBSEM image volume at $10 \times 10 \times 10$ nm/pixel resolution. We divide the whole volume evenly into eight $250 \times 250 \times 250$ subvolumes and do eight-fold cross validation using one subvolume each time as the supervised set and the whole volume as the testing data. Each subvolume has from 204 to 260 ground truth neuron segments that are larger than 100 pixels. Following the setting in the mouse cortex data set experiment, we use subsets of 100%, 50%, 25%, 12.5%, 6.25%, and 3.125% of all true neuron segments from the respective supervised subvolume in each fold of the cross validation as the supervised data to generate boundary classification labels. We use the entire volume to generate unsupervised samples. We target 3D segmentation for this data set.

5.3.2 Experiments

To generate initial superpixels, we use the watershed algorithm [64] over the membrane detection confidence maps generated using CHM [66]. For the boundary classifier, we use features including shape information (region size, perimeter, bounding box, boundary length, etc.) and image intensity statistics (mean, standard deviation, minimum, maximum, etc.) of region interior and boundary pixels from both the original EM images and membrane detection confidence maps.

We use the adapted Rand F-error metric [39] to generate boundary classification labels using whole ground truth images (Section 2.2.3) for the 2D mouse neuropil data set. For the 3D mouse cortex and *Drosophila* neuropil data set, we determine the labels using individual ground truth segments instead. We use this setting in order to match the actual process of analyzing EM images by neuroscientists. Details about label generation using individual ground truth segments are provided in Appendix B.

We can see in (5.2) and (5.8) that computing $\tilde{F}_i^L$ and its gradient involves multiplications of L floating point numbers, which can cause underflow problems for leaf-to-root clique paths in a merge tree of even moderate height. To avoid this problem, we exploit the local

property of the merge consistency constraint and compute $\tilde{F}_i^L$ for every path subpart of small length L. In this work, we use $L = 3$ for all experiments. For inference, we initialize $\boldsymbol{w}$ by running gradient descent on (5.5) with only the supervised term and the regularizer before adding the unsupervised term for the whole optimization. We update σ_u and σ_s in between every 100 gradient descent steps on $\boldsymbol{w}$.

We compare SSHMT with the fully supervised HMT [41] as the baseline method. To make the comparison fair, we use the same logistic sigmoid function as the boundary classifier for both HMT and SSHMT. The fully supervised training uses the same Bayesian framework only without the unsupervised term in (5.5) and alternately estimates σ_s to balance the regularization term and the supervised term. All the hyperparameters are kept identical for HMT and SSHMT and fixed for all experiments. We use the adapted Rand F-error [39] following the public EM image segmentation challenges [68, 70]. Due to the randomness in the selection of supervised data, we repeat each experiment 50 times, except in the cases in which there are fewer possible combinations. We report the mean and standard deviation of testing errors for each set of repeats on the three data sets in Table 5.1.

Examples of 2D segmentation testing results from the mouse neuropil data set using fully supervised HMT and SSHMT with 1 (7.143%) ground truth image as supervised data are shown in Figure 5.1. Examples of 3D individual neuron segmentation testing results from the *Drosophila* neuropil data set using fully supervised HMT and SSHMT with 12 (6.25%) true neuron segments as supervised data are visualized using TrakEM2 [38] in Fiji [73] and shown in Figure 5.2.

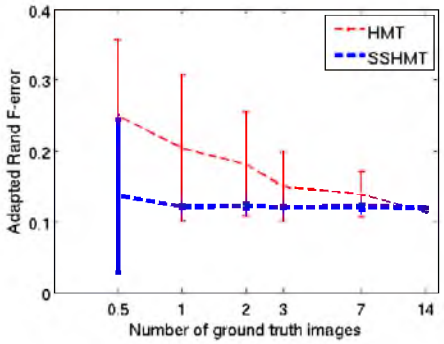
From Table 5.1, we can see that with abundant supervised data, the performance of SSHMT is similar to HMT in terms of segmentation accuracy. When the amount of supervised data becomes smaller, SSHMT significantly outperforms the fully supervised method with accuracy close to the HMT results using the full supervised sets. Moreover, the introduction of the unsupervised term stabilizes the learning of the classification function and results in much more consistent segmentation performance, even when only very limited (3% to 7%) label data are available. Increases in errors and large variations are observed in the SSHMT results when the supervised data become too scarce, because the few supervised samples are incapable of providing sufficient guidance to balance the unsupervised term, and the boundary classifiers are biased to give trivial predictions.

Figure 5.1 shows that SSHMT is capable of fixing both over- and undersegmentation errors that occur in the HMT results. Figure 5.2 also shows that SSHMT can fix over-

Table 5.1: Means and standard deviations of the adapted Rand F-errors of HMT and SSHMT segmentations for the three EM data sets. The left table columns show the amount of used ground truth data, in terms of (a) the number of images, (b) the number of segments, and (c) the percentage of all segments. Bold numbers in the tables show the results of the higher accuracy under comparison. The figures on the right visualize the means (dashed lines) and the standard deviations (solid bars) of the errors of HMT (red) and SSHMT (blue) results for each data set.

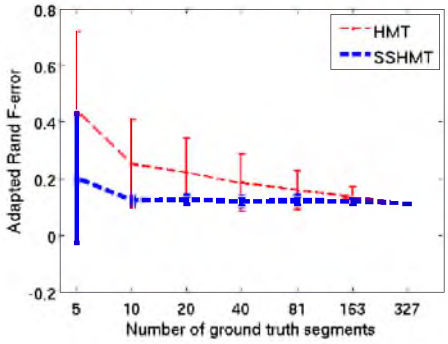
(a) Mouse neuropil

	HMT		SSHMT	
#GT	Mean	Std.	Mean	Std.
14	**0.1135**	-	0.1196	-
7	0.1382	0.03238	**0.1208**	0.004033
3	0.1492	0.04851	**0.1205**	0.001383
2	0.1811	0.07346	**0.1217**	0.004116
1	0.2035	0.1029	**0.1210**	0.002206
0.5	0.2505	0.1062	**0.1365**	0.1079



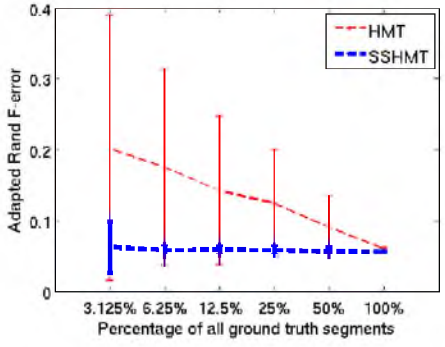
(b) Mouse cortex

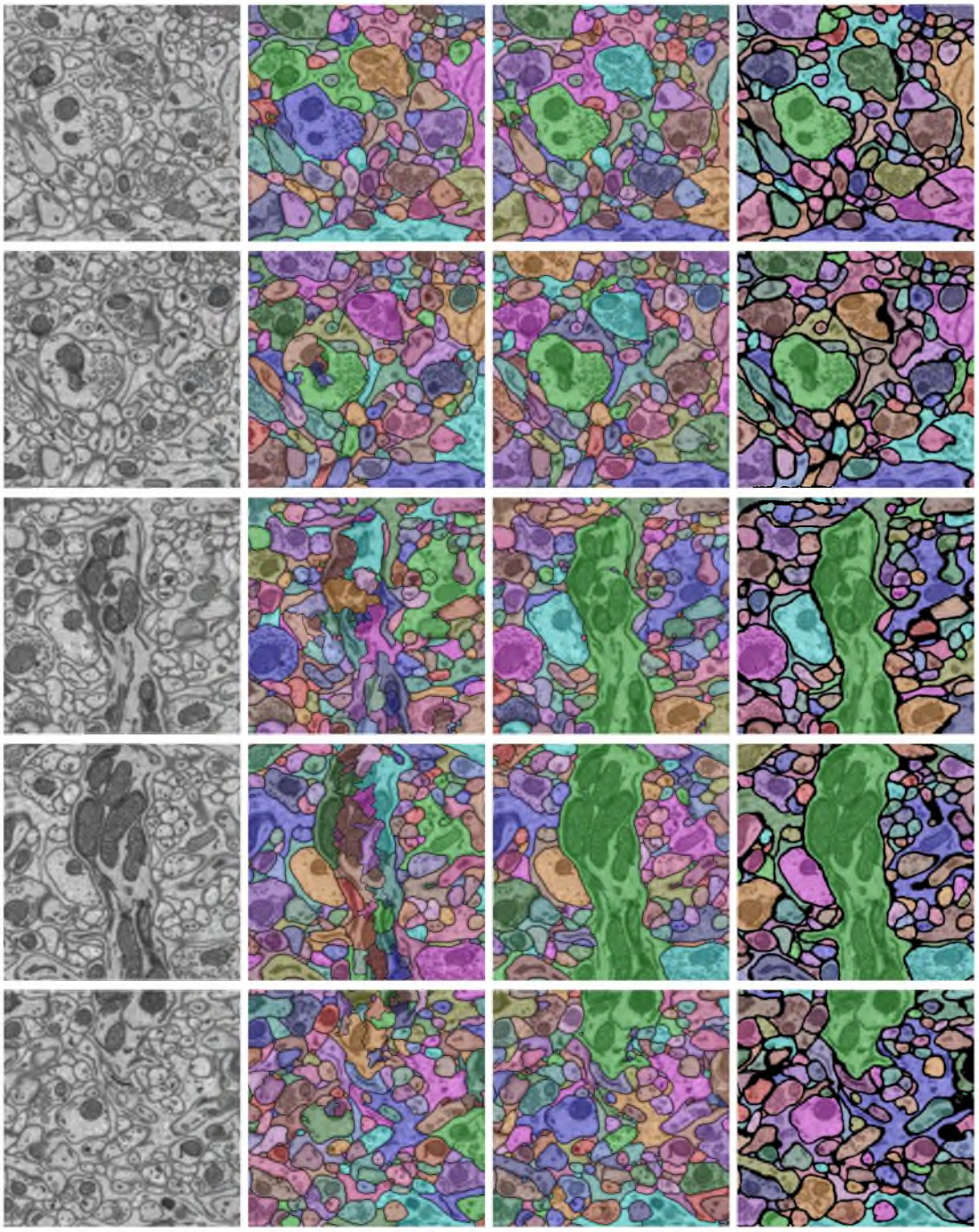
	HMT		SSHMT	
#GT	Mean	Std.	Mean	Std.
327	**0.1101**	-	0.1104	-
163	0.1344	0.03660	**0.1189**	0.01506
81	0.1583	0.06909	**0.1215**	0.01661
40	0.1844	0.1019	**0.1198**	0.01690
20	0.2205	0.1226	**0.1238**	0.01466
10	0.2503	0.1561	**0.1219**	0.01273
5	0.4389	0.2769	**0.2008**	0.2285



(c) *Drosophila* neuropil

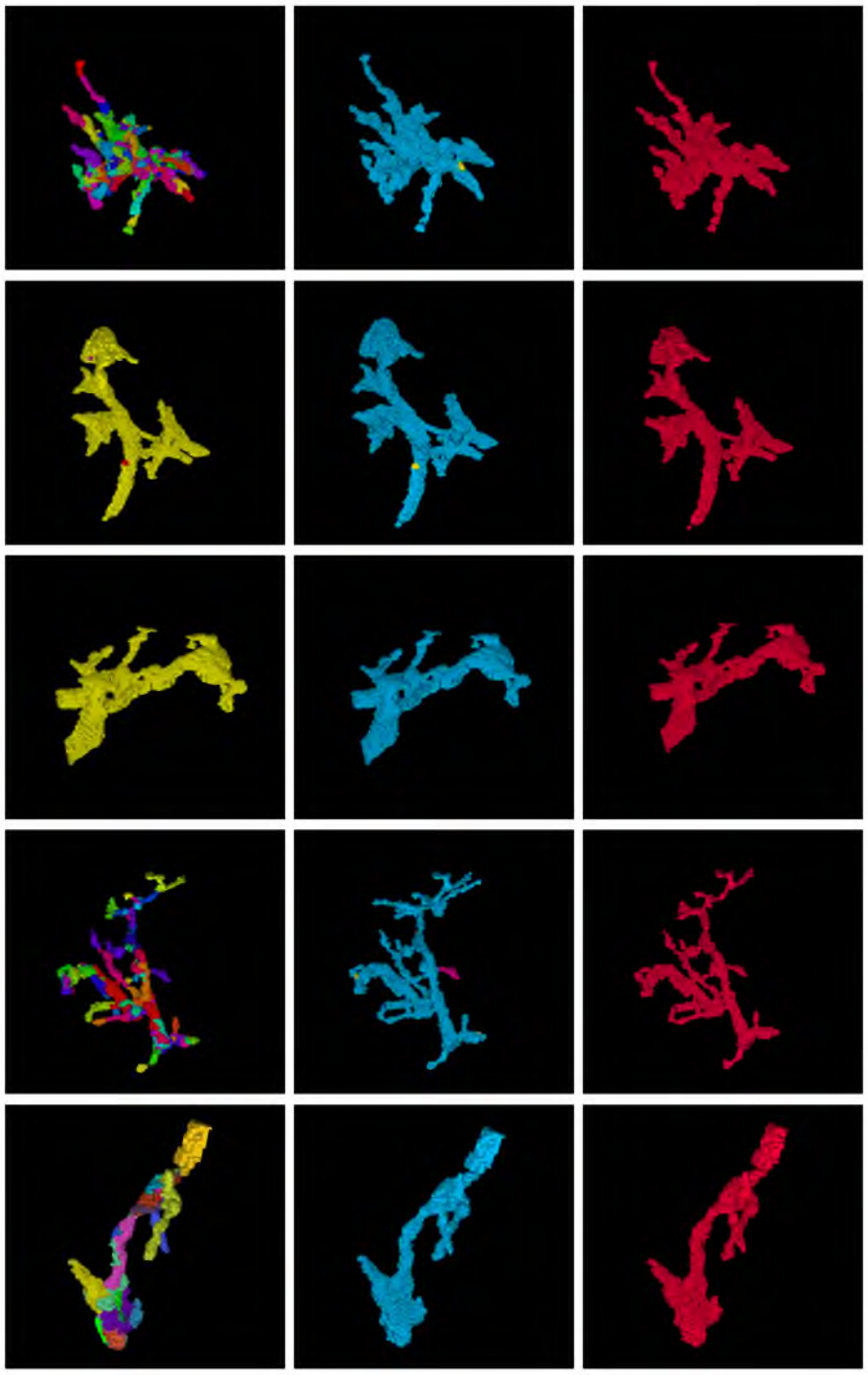
	HMT		SSHMT	
%GT	Mean	Std.	Mean	Std.
100%	0.06044	-	**0.05504**	-
50%	0.09004	0.04476	**0.05602**	0.005550
25%	0.1240	0.07491	**0.05803**	0.007703
12.5%	0.1418	0.1055	**0.05835**	0.007797
6.25%	0.1748	0.1389	**0.05756**	0.008933
3.125%	0.2017	0.1871	**0.06213**	0.03660





(a) Original (b) HMT (c) SSHMT (d) Ground truth

Figure 5.1: Examples of 2D segmentation results (five sections) of the mouse neuropil data set, including (in columns) (a) original EM images, segmentation results using (b) HMT and (c) SSHMT with one ground truth image as supervised data, and (d) the corresponding ground truth images. Different colors indicate individual segments.



(a) HMT (b) SSHMT (c) Ground truth

Figure 5.2: Examples of five individual neurons from 3D segmentation results of the *Drosophila* neuropil data set, including segmentation results (in columns) using (a) HMT and (b) SSHMT with 12 (6.25%) 3D ground truth segments as supervised data, and (c) the corresponding ground truth segments. Different colors indicate individual segments.

segmentation errors and generate highly accurate neuron segmentations. Note that in our experiments, we always randomly select the supervised data subsets. For realistic uses, we expect supervised samples of better representativeness to be provided with expertise and the performance of SSHMT to be further improved.

5.4 Conclusion

In this chapter, we proposed a semi-supervised method that can consistently learn boundary classifiers with a very limited amount of supervised data for region-based image segmentation. This method dramatically reduces the high demands for ground truth data by fully supervised algorithms. We applied our method to neuron segmentation in EM images from three data sets and demonstrated that by using only a small amount of ground truth data, our method performs close to the state-of-the-art fully supervised method with full labeled data sets. In our future work, we will explore the integration of the proposed constraint-based unsupervised loss in structural learning settings to further exploit the structured information for learning the boundary classification function. Also, we may replace the current logistic sigmoid function with more complex classifiers and combine our method with active learning frameworks to improve segmentation accuracy.

CHAPTER 6

DISCUSSION AND FUTURE WORK

This dissertation focused on the development of learning-based hierarchical models for image region segmentation. Our work was motivated by the segmentation problem of neuronal structures in EM images required by connectomics research. We proposed the HMT model and the HMF model for EM image segmentation. We also extended the HMT model to solve general image segmentation problems. We demonstrated that the HMT model and its variants achieve state-of-the-art performance on segmenting both types of images. We also proposed a semi-supervised learning algorithm that can be used based on the HMT model to achieve competitive segmentation performance with only very limited labeled data.

As the center of our work, the HMT model uses a tree structure to represent the hierarchy of region merging and provides a learning-based framework for efficient image segmentation. The main advantage of using the merge tree structure is three-fold. First, compared with modeling segmentation as a planar graph problem, richer features can be effectively extracted from larger regions instead of using only features from initial superpixels. Second, different from greedy region agglomeration methods that make merging decisions locally, the final segmentation inference in the HMT model can be done with consideration of the whole merging process, which effectively reduces the impact of inaccurate local predictions to the entire segmentation result. Third, thanks to the tree structure, both greedy and globally optimal inference can be done efficiently in polynomial time. In other words, the HMT model provides a balanced solution to the trade-off between utilization of higher order information and tractability. The HMT model itself requires no parameter that needs manually tuning. Also, it makes no assumption about image dimensionality and can be applied to solving multidimensional image segmentation problems without any change. Furthermore, as a generic framework, the HMT model can work for any type of merging-based decision-making problem as long as the combination of separate objects can be defined and its likelihood can be quantified.

In the future work, there are unresolved and new questions we would like to answer:

1. *Can multiple trees for the same image be jointly used to improve segmentation?* One major concern about the merge structure is it is not able to fix incorrect merging orders. This motivated the development of the iterative merge sampling algorithm (Section 4.2.4) for natural image segmentation, and we have shown that the segmentation accuracy of the final contour hierarchy can be significantly improved by averaging individual segmentation from each tree. In the case of single-shot segmentation that targets generating one hard label map per image, such as EM image segmentation, it would be interesting to see if multiple tree structures for a single image can be combined for joint inference of the final segmentation. Following this idea, we have worked on developing a greedy optimization algorithm, similar to the one in HMT (Section 2.2.4), that iteratively selects the best node in the multiple trees and removes all the nodes in each tree that share identical initial superpixels with the selected node until every node is processed. We have also developed a globally optimal inference algorithm similar to the bottom-up/top-down algorithm (Section 2.2.4), except that the energy tuples are propagated such that the energy for each case is minimized across all trees. We did not draw conclusive results from both algorithms and will investigate the reason.

2. *Can the boundary detection/region segmentation pipeline be unified into a joint model?* The commonly used boundary detection/region segmentation pipeline has its problem that the two steps essentially target different goals. A pixelwise boundary detector is trained to reduce pixel error, which can be suboptimal in achieving accurate region segmentation. Also, it uses image context within a fixed window around the pixel, which can be inefficient in incorporating more targeted region information, especially when different regions have large variations in size such as neuron cells in EM images. On the other hand, current region-based methods, such as the HMT model, usually use boundary detection confidence as a very important indication about region merging. Thus, they can be severely affected by boundary mistakes and are not able to recover from large false positive or false negative boundary detections. In this sense, developing a joint model that incorporates both the two steps and unifies their targets would be of great interest. Following this idea, we have conducted a preliminary work that incorporates a type of tree-derived pixel loss in training the CHM algorithm to generate boundary confidence maps that are less prone to causing incorrect merging

orders, but the results are inconclusive. We will further look into this problem in our future work.

3. *Can the proposed hierarchical frameworks be applied in other applications?* It would be interesting to see if and how the proposed HMT framework and its variants can be applied in other applications. For example, the HMT model is designed for full image segmentation, and it needs to be modified to solve foreground/background segmentation problems, such as segmentation of intracellular structures (e.g., mitochondria) in EM images or of specific objects in natural images. In addition, we may be able to apply the HMF model to segmenting consecutive image sequences, such as videos. The SSHMT model can be generalized for learning to either segment or detect under the tree structure. We are working on applying it to cell detection using other imaging modalities.

APPENDIX A

SUMMARY OF CLASSIFICATION FEATURES

A.1 Boundary Classification Features for EM Image Segmentation

The categories of features used for boundary classification for EM images (Section 2.2.3) are summarized as follows. Note that for the texton features, 7×7 patches are used and k-means clustering is used for learning the texture dictionary of 100 words. Specific feature choices may differ in different implementations.

1. Shape: Region areas, perimeters, and compactness; boundary length and curvatures (2D only).

2. Image appearance (of original EM images and membrane detection confidence maps): Boundary pixel intensity histogram (10-bin) and statistics; region pixel intensity histogram (10-bin) and statistics; region texton histogram (100-bin, 2D only).

3. Merging saliencies.

A.2 Section Classification Features for EM Image Segmentation

The categories of features used for section classification for EM images (Section 3.2.2) are summarized as follows. Note that for the texton features, 7×7 patches are used and k-means clustering is used for learning the texture dictionary of 100 words. Specific feature choices may differ in different implementations.

1. Shape: Region areas, perimeters, compactness, centroid distance, and overlap.

2. Image appearance (of original EM images and membrane detection confidence maps): Boundary pixel intensity histogram (10-bin) and statistics; region pixel intensity histogram (10-bin) and statistics; region texton histogram (100-bin).

A.3 Boundary Classification Features for Natural Image Segmentation

We use 55 features from region pairs to train the boundary classifiers, including:

1. Geometry (5-dimensional): Areas of two regions normalized by image area and perimeters and boundary length of two regions normalized by the length of the image diagonal.

2. Boundary (4-dimensional): Means and medians of boundary pixel intensities from gPb and UCM [5]. Boundary detector gPb generates probability maps that describe how likely each pixel belongs to an image boundary. UCM is the result from postprocessing gPb probability maps that depicts how boundary pixels contribute to contour hierarchies in images. The boundary pixels follow the definition in (2.2).

3. Color (24-dimensional): Absolute mean differences, L_1 and χ^2 distances, and absolute entropy differences between histograms (10-bin) of LAB and HSV components of the original images.

4. Texture (8-dimensional): L_1 and χ^2 distances between histograms of texton [114] (64-bin) and SIFT [115] dictionary of 256 words. The SIFT descriptors are computed densely, and 8×8 patches are used on gray, A, and B channels of the original images.

5. Geometric context (14-dimension): L_1 and χ^2 distances between histograms (32-bin) of the probability maps of each of the seven geometric context labels. The geometric context labels indicate orientations of the surfaces in the images, which are predicted by a fixed pretrained model provided by [116].

APPENDIX B

BOUNDARY CLASSIFICATION LABEL GENERATION USING INDIVIDUAL GROUND TRUTH SEGMENTS

Assume we have only individual annotated image segments instead of entire image volumes as ground truth. Given a merge tree, we generate the best-effort ground truth classification labels for a subset of cliques as follows:

1. For every region represented by a tree node, compute the Jaccard indices of this region against all the annotated ground truth segments. Use the highest Jaccard index of each node as its eligible score.

2. Mark every node in the tree as "eligible" if its eligible score is above a certain threshold (0.75 in practice) or "ineligible" otherwise.

3. Iteratively select a currently "eligible" node with the highest eligible score; mark it and its ancestors and descendants as "ineligible," until every node is "ineligible." This procedure generates a set of selected nodes.

4. For every selected node, label the cliques at itself and its descendants as $y = 1$ ("merge") and the cliques at its ancestors as $y = 0$ ("split").

Eventually, the clique samples that receive merge/split labels are considered to be the supervised data.

REFERENCES

[1] C. Gu, J. J. Lim, P. Arbeláez, and J. Malik, "Recognition using regions," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2009, pp. 1030–1037.

[2] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[3] J. C. Gee, M. Reivich, and R. Bajcsy, "Elastically deforming 3D atlas to match anatomical brain images," *J. Comput. Assisted Tomogr.*, vol. 17, no. 2, pp. 225–236, 1993.

[4] M. Helmstaedter, K. L. Briggman, S. C. Turaga, V. Jain, H. S. Seung, and W. Denk, "Connectomic reconstruction of the inner plexiform layer in the mouse retina," *Nature*, vol. 500, no. 7461, pp. 168–174, 2013.

[5] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.

[6] S. Belongie, C. Carson, H. Greenspan, and J. Malik, "Color- and texture-based image segmentation using EM and its application to content-based image retrieval," in *Proc. Int. Conf. Comput. Vision*, 1998, pp. 675–682.

[7] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[8] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[9] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. Int. Conf. Comput. Vision*, 2003, pp. 10–17.

[10] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, pp. 721–741, 1984.

[11] D. Greig, B. Porteous, and A. H. Seheult, "Exact maximum a posteriori estimation for binary images," *J. Roy. Stat. Soc., Series B, Methodological*, pp. 271–279, 1989.

[12] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.

[13] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[14] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis.* Wiley New York, 1973, vol. 3.

[15] O. Sporns, G. Tononi, and R. Kötter, "The human connectome: a structural description of the human brain," *PLoS Comput. Biol.*, vol. 1, no. 4, p. e42, 2005.

[16] K. L. Briggman, M. Helmstaedter, and W. Denk, "Wiring specificity in the direction-selectivity circuit of the retina," *Nature*, vol. 471, no. 7337, pp. 183–188, 2011.

[17] D. D. Bock, W.-C. A. Lee, A. M. Kerlin, M. L. Andermann, G. Hood, A. W. Wetzel, S. Yurgenson, E. R. Soucy, H. S. Kim, and R. C. Reid, "Network anatomy and in vivo physiology of visual cortical neurons," *Nature*, vol. 471, no. 7337, pp. 177–182, 2011.

[18] H. S. Seung, "Neuroscience: towards functional connectomics," *Nature*, vol. 471, no. 7337, pp. 170–172, 2011.

[19] Y.-W. Peng, Y. Hao, R. M. Petters, and F. Wong, "Ectopic synaptogenesis in the mammalian retina caused by rod photoreceptor-specific mutations," *Nature Neurosci.*, vol. 3, no. 11, pp. 1121–1127, 2000.

[20] R. E. Marc, B. W. Jones, C. B. Watt, and E. Strettoi, "Neural remodeling in retinal degeneration," *Prog. Retinal Eye Res.*, vol. 22, no. 5, pp. 607–655, 2003.

[21] B. W. Jones, C. B. Watt, J. M. Frederick, W. Baehr, C.-K. Chen, E. M. Levine, A. H. Milam, M. M. Lavail, and R. E. Marc, "Retinal remodeling triggered by photoreceptor degenerations," *J. Comp. Neurol.*, vol. 464, no. 1, pp. 1–16, 2003.

[22] B. W. Jones and R. E. Marc, "Retinal remodeling during retinal degeneration," *Exp. Eye Res.*, vol. 81, no. 2, pp. 123–137, 2005.

[23] J. R. Anderson, B. W. Jones, J.-H. Yang, M. V. Shaw, C. B. Watt, P. Koshevoy, J. Spaltenstein, E. Jurrus, U. Kannan, R. T. Whitaker *et al.*, "A computational framework for ultrastructural mapping of neural circuitry," *PLoS Biol.*, vol. 7, no. 3, p. e1000074, 2009.

[24] D. B. Chklovskii, S. Vitaladevuni, and L. K. Scheffer, "Semi-automated reconstruction of neural circuits using electron microscopy," *Cur. Opin. Neurobiol.*, vol. 20, no. 5, pp. 667–675, 2010.

[25] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS Biol.*, vol. 2, no. 11, p. e329, 2004.

[26] H. Horstmann, C. Körber, K. Sätzler, D. Aydin, and T. Kuner, "Serial section scanning electron microscopy (S3EM) on silicon wafers for ultra-structural volume imaging of cells and tissues," *PLoS One*, vol. 7, no. 4, p. e35172, 2012.

[27] G. Knott, H. Marchman, D. Wall, and B. Lich, "Serial section scanning electron microscopy of adult brain tissue using focused ion beam milling," *J. Neurosci.*, vol. 28, no. 12, pp. 2959–2964, 2008.

[28] J. R. Anderson, B. W. Jones, C. B. Watt, M. V. Shaw, J.-H. Yang, D. DeMill, J. S. Lauritzen, Y. Lin, K. D. Rapp, D. Mastronarde, P. Koshevoy, B. Grimm, T. Tasdizen, R. Whitaker, and R. E. Marc, "Exploring the retinal connectome," *Mol. Vision*, vol. 17, p. 355, 2011.

[29] K. L. Briggman and W. Denk, "Towards neural circuit reconstruction with volume electron microscopy techniques," *Cur. Opin. Neurobiol.*, vol. 16, no. 5, pp. 562–570, 2006.

[30] V. Jain, J. F. Murray, F. Roth, S. Turaga, V. Zhigulin, K. L. Briggman, M. N. Helmstaedter, W. Denk, and H. S. Seung, "Supervised learning of image restoration with convolutional networks," in *Proc. Int. Conf. Comput. Vision*, 2007, pp. 1–8.

[31] A. Lucchi, K. Smith, R. Achanta, V. Lepetit, and P. Fua, "A fully automated approach to segmentation of irregularly shaped cellular structures in EM images," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2010, pp. 463–471.

[32] S. Borra and S. Sarkar, "A framework for performance characterization of intermediate-level grouping modules," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 11, pp. 1306–1312, 1997.

[33] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. Int. Conf. Comput. Vision*, vol. 2, 2001, pp. 416–423.

[34] T. S. Yoo, M. J. Ackerman, W. E. Lorensen, W. Schroeder, V. Chalana, S. Aylward, D. Metaxas, and R. Whitaker, "Engineering and algorithm design for an image processing API: a technical report on ITK—the Insight Toolkit," *Studies in Health Technol. and Inform.*, pp. 586–592, 2002.

[35] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.

[36] B. Dawes, D. Abrahams *et al.*, "Boost C++ Libraries," http://www.boost.org, 2000.

[37] A. Cardona, S. Saalfeld, S. Preibisch, B. Schmid, A. Cheng, J. Pulokas, P. Tomancak, and V. Hartenstein, "An integrated micro-and macroarchitectural analysis of the *Drosophila* brain by computer-assisted serial section electron microscopy," *PLoS Biol.*, vol. 8, no. 10, p. e1000502, 2010.

[38] A. Cardona, S. Saalfeld, J. Schindelin, I. Arganda-Carreras, S. Preibisch, M. Longair, P. Tomancak, V. Hartenstein, and R. J. Douglas, "TrakEM2 software for neural circuit reconstruction," *PLoS One*, vol. 7, no. 6, p. e38011, 2012.

[39] I. Arganda-Carreras, S. C. Turaga, D. R. Berger, D. Cireşan, A. Giusti, L. M. Gambardella, J. Schmidhuber, D. Laptev, S. Dwivedi, J. M. Buhmann *et al.*, "Crowdsourcing the creation of image segmentation algorithms for connectomics," *Front. Neuroanat.*, vol. 9, 2015.

[40] T. Liu, E. Jurrus, M. Seyedhosseini, M. Ellisman, and T. Tasdizen, "Watershed merge tree classification for electron microscopy image segmentation," in *Proc. Int. Conf. Pattern Recognition*, 2012, pp. 133–137.

[41] T. Liu, C. Jones, M. Seyedhosseini, and T. Tasdizen, "A modular hierarchical approach to 3D electron microscopy image segmentation," *J. Neurosci. Methods*, vol. 226, pp. 88–102, 2014.

[42] T. Tasdizen, R. Whitaker, R. Marc, and B. Jones, "Enhancement of cell boundaries in transmission electron microscopy images," in *Proc. Int. Conf. Image Process.*, vol. 2, 2005, pp. II–129.

[43] E. Jurrus, R. Whitaker, B. W. Jones, R. E. Marc, and T. Tasdizen, "An optimal-path approach for neural circuit reconstruction," in *Proc. Int. Symp. Biomed. Imag.*, 2008, pp. 1609–1612.

[44] R. Kumar, A. Vázquez-Reina, and H. Pfister, "Radon-like features and their application to connectomics," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition Workshops*, 2010, pp. 186–193.

[45] Y. Mishchenko, "Automation of 3D reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs," *J. Neurosci. Methods*, vol. 176, no. 2, pp. 276–289, 2009.

[46] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Adv. Neural Inform. Process. Syst.*, 2012, pp. 2852–2860.

[47] K. Lee, A. Zlateski, V. Ashwin, and H. S. Seung, "Recursive training of 2D-3D convolutional networks for neuronal boundary prediction," in *Adv. Neural Inform. Process. Syst.*, 2015, pp. 3559–3567.

[48] M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber, "Parallel multi-dimensional LSTM, with application to fast biomedical volumetric image segmentation," in *Adv. Neural Inform. Process. Syst.*, 2015, pp. 2980–2988.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2015, pp. 234–241.

[50] D. Laptev, A. Vezhnevets, S. Dwivedi, and J. M. Buhmann, "Anisotropic ssTEM image segmentation using dense correspondence across sections," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2012, pp. 323–330.

[51] M. Seyedhosseini and T. Tasdizen, "Multi-class multi-scale series contextual model for image segmentation," *IEEE Trans. Image Process.*, vol. 22, no. 11, pp. 4486–4496, 2013.

[52] V. Kaynig, T. Fuchs, and J. M. Buhmann, "Neuron geometry extraction by perceptual grouping in ssTEM images," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2010, pp. 2902–2909.

[53] H.-F. Yang and Y. Choe, "Cell tracking and segmentation in electron microscopy images using graph cuts," in *Proc. Int. Symp. Biomed. Imag.*, 2009, pp. 306–309.

[54] V. Kaynig, T. J. Fuchs, and J. M. Buhmann, "Geometrical consistent 3D tracing of neuronal processes in ssTEM data," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2010, pp. 209–216.

[55] S. N. Vitaladevuni and R. Basri, "Co-clustering of image segments using convex optimization applied to EM neuronal reconstruction," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2010, pp. 2203–2210.

[56] B. Andres, U. Koethe, T. Kroeger, M. Helmstaedter, K. L. Briggman, W. Denk, and F. A. Hamprecht, "3D segmentation of SBFSEM images of neuropil by a graphical model over supervoxel boundaries," *Med. Image Anal.*, vol. 16, no. 4, pp. 796–805, 2012.

[57] A. Vazquez-Reina, M. Gelbart, D. Huang, J. Lichtman, E. Miller, and H. Pfister, "Segmentation fusion for connectomics," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 177–184.

[58] J. Funke, B. Andres, F. A. Hamprecht, A. Cardona, and M. Cook, "Efficient automatic 3D-reconstruction of branching neurons from EM data," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2012, pp. 1004–1011.

[59] V. Jain, S. C. Turaga, K. Briggman, M. N. Helmstaedter, W. Denk, and H. S. Seung, "Learning to agglomerate superpixel hierarchies," in *Adv. Neural Inform. Process. Syst.*, 2011, pp. 648–656.

[60] J. Nunez-Iglesias, R. Kennedy, T. Parag, J. Shi, and D. B. Chklovskii, "Machine learning of hierarchical clustering to segment 2D and 3D images," *PLoS One*, vol. 8, no. 8, p. e71715, 2013.

[61] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2290–2297, 2009.

[62] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2010, pp. 211–224.

[63] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, 2012.

[64] S. Beucher and C. Lantuejoul, "Use of watersheds in contour detection," *Int. Workshop Image Process.: Real-time Edge and Motion Detection/Estimation*, 1979.

[65] R. Beare and G. Lehmann, "The watershed transform in ITK—discussion and new developments," *Insight J.*, January - June 2006.

[66] M. Seyedhosseini, M. Sajjadi, and T. Tasdizen, "Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks," in *Proc. Int. Conf. Comput. Vision*, 2013, pp. 2168–2175.

[67] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[68] I. Arganda-Carreras, H. S. Seung, A. Cardona, and J. Schindelin, "Segmentation of neuronal structures in EM stacks challenge - ISBI 2012," http://brainiac2.mit.edu/isbi_challenge/, 2012, accessed on March 29, 2016.

[69] T. J. Deerinck, E. A. Bushong, V. Lev-Ram, X. Shu, R. Y. Tsien, and M. H. Ellisman, "Enhancing serial block-face scanning electron microscopy to enable high resolution 3-D nanohistology of cells and tissues," *Microsc. Microanal.*, vol. 16, no. S2, pp. 1138–1139, 2010.

[70] I. Arganda-Carreras, H. S. Seung, A. Vishwanathan, and D. Berger, "3D Segmentation of Neurites in EM Images Challenge - ISBI 2013," http://brainiac2.mit.edu/SNEMI3D, 2013, accessed on March 29, 2016.

[71] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Stat. Assoc.*, vol. 66, no. 336, pp. 846–850, 1971.

[72] M. Sajjadi, M. Seyedhosseini, and T. Tasdizen, "Disjunctive normal networks," *arXiv preprint arXiv:1412.8534*, 2014.

[73] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid *et al.*, "Fiji: an open-source platform for biological-image analysis," *Nature Methods*, vol. 9, no. 7, pp. 676–682, 2012.

[74] T. Liu, M. Seyedhosseini, M. Ellisman, and T. Tasdizen, "Watershed merge forest classification for electron microscopy image stack segmentation," in *Proc. Int. Conf. Image Process.*, 2013, pp. 4069–4073.

[75] C. Jones, T. Liu, N. W. Cohan, M. Ellisman, and T. Tasdizen, "Efficient semi-automatic 3D segmentation for neuron tracing in electron microscopy images," *J. Neurosci. Methods*, vol. 246, pp. 13–21, 2015.

[76] A. Joulin, F. Bach, and J. Ponce, "Discriminative clustering for image co-segmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2010, pp. 1943–1950.

[77] S. Vicente, C. Rother, and V. Kolmogorov, "Object cosegmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2011, pp. 2217–2224.

[78] G. Kim, E. P. Xing, L. Fei-Fei, and T. Kanade, "Distributed cosegmentation via submodular optimization on anisotropic diffusion," in *Proc. Int. Conf. Comput. Vision*, 2011, pp. 169–176.

[79] A. Jaiantilal, "Classification and regression by randomforest-matlab," Available at http://code.google.com/p/randomforest-matlab, 2009.

[80] T. Liu, M. Seyedhosseini, and T. Tasdizen, "Image segmentation using hierarchical merge tree," *arXiv preprint arXiv:1505.06389*, 2015.

[81] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Roy. Soc. London, Series B, Biological Sciences*, 1980.

[82] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, pp. 679–698, 1986.

[83] M. C. Morrone and R. A. Owens, "Feature detection from local energy," *Pattern Recognition Lett.*, vol. 6, no. 5, pp. 303–313, 1987.

[84] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 9, pp. 891–906, 1991.

[85] X. Ren and L. Bo, "Discriminatively trained sparse code gradients for contour detection," in *Adv. Neural Inform. Process. Syst.*, 2012, pp. 584–592.

[86] P. Dollár and C. Zitnick, "Structured forests for fast edge detection," in *Proc. Int. Conf. Comput. Vision*, 2013, pp. 1841–1848.

[87] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2008, pp. 705–718.

[88] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2006, pp. 1–15.

[89] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *Proc. Int. Conf. Comput. Vision*, 2009, pp. 1–8.

[90] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[91] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2012, pp. 746–760.

[92] S. Kim, C. D. Yoo, S. Nowozin, and P. Kohli, "Image segmentation using higher-order correlation clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 9, pp. 1761–1774, 2014.

[93] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2014, pp. 328–335.

[94] Y. Yu, C. Fang, and Z. Liao, "Piecewise flat embedding for image segmentation," in *Proc. Int. Conf. Comput. Vision*, 2015, pp. 1368–1376.

[95] Z. Ren and G. Shakhnarovich, "Image segmentation by cascaded region agglomeration," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2013, pp. 2011–2018.

[96] E. Borenstein and S. Ullman, "Combined top-down/bottom-up segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 12, pp. 2109–2125, 2008.

[97] J. Carreira and C. Sminchisescu, "CPMC: Automatic object segmentation using constrained parametric min-cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1312–1328, 2012.

[98] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik, "Semantic segmentation using regions and parts," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2012, pp. 3378–3385.

[99] D. Weiss and B. Taskar, "SCALPEL: Segmentation cascades with localized priors and efficient learning," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2013, pp. 2035–2042.

[100] M. Meil, "Comparing clusterings: an axiomatic view," in *Proc. Int. Conf. Mach. Learn.*, 2005, pp. 577–584.

[101] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry, "Unsupervised segmentation of natural images via lossy data compression," *Comput. Vision Image Understanding*, vol. 110, no. 2, pp. 212–225, 2008.

[102] J. Pearl, "Reverend bayes on inference engines: A distributed hierarchical approach," in *Proc. AAAI Conf. Artif. Intell.*, 1982, pp. 133–136.

[103] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[104] T. Malisiewicz and A. A. Efros, "Improving spatial support for objects via multiple segmentations," in *Proc. Brit. Mach. Vision Conf.*, 2007.

[105] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 929–944, 2007.

[106] M. Donoser and D. Schmalstieg, "Discrete-continuous gradient orientation estimation for faster image segmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2014, pp. 3158–3165.

[107] C. Sommer, C. Straehle, U. Koethe, and F. A. Hamprecht, "ilastik: Interactive learning and segmentation toolkit," in *Proc. Int. Symp. Biomed. Imag.*, 2011, pp. 230–233.

[108] V. Kaynig, A. Vazquez-Reina, S. Knowles-Barley, M. Roberts, T. R. Jones, N. Kasthuri, E. Miller, J. Lichtman, and H. Pfister, "Large-scale automatic reconstruction of neuronal processes from electron microscopy images," *Med. Image Anal.*, vol. 22, no. 1, pp. 77–88, 2015.

[109] N. Krasowski, T. Beier, G. Knott, U. Koethe, F. Hamprecht, and A. Kreshuk, "Improving 3D EM data segmentation by joint optimization over boundary evidence and biological priors," in *Proc. Int. Symp. Biomed. Imag.*, 2015, pp. 536–539.

[110] J. Funke, F. A. Hamprecht, and C. Zhang, "Learning to segment: Training hierarchical segmentation under a topological loss," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2015, pp. 268–275.

[111] M. G. Uzunbas, C. Chen, and D. Metaxas, "An efficient conditional random field approach for automatic and interactive neuron segmentation," *Med. Image Anal.*, vol. 27, pp. 31–44, 2016.

[112] T. Parag, S. Plaza, and L. Scheffer, "Small sample learning of superpixel classifiers for EM segmentation," in *Med. Image Comput. Comput.-Assisted Intervention*. Springer, 2014, pp. 389–397.

[113] T. Parag, D. C. Ciresan, and A. Giusti, "Efficient classifier training to minimize false merges in electron microscopy segmentation," in *Proc. Int. Conf. Comput. Vision*, 2015, pp. 657–665.

[114] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *Int. J. Comput. Vision*, vol. 43, no. 1, pp. 7–27, 2001.

[115] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Comput. Vision*, vol. 2, 1999, pp. 1150–1157.

[116] D. Hoiem, A. A. Efros, and M. Hebert, "Geometric context from a single image," in *Proc. Int. Conf. Comput. Vision*, vol. 1, 2005, pp. 654–661.